BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

FIG. 1A

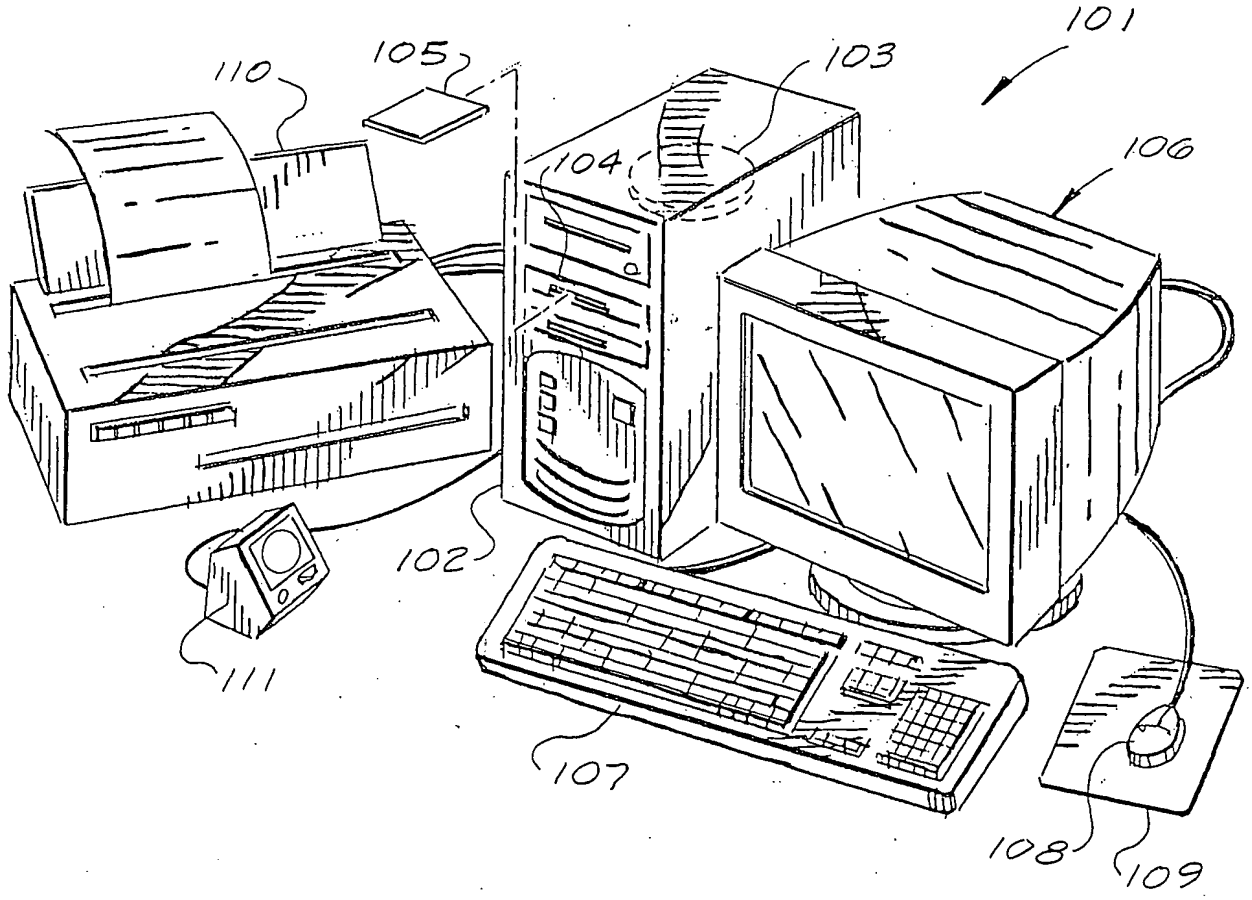


FIG. 1B-1

General Listing and Description of Primary ADS Data Types

In - Input device n of the ADS
Xn - sensory "reading" of input device In (may be proprioceptive)
XRFn - a specified range/purview of In
RF - a rule that is "true" if Xn is included in XRFn
BF - a set of RF's identifying a "doing"
RFs - RF about the self ADS
RFsB - an RFs identifying a doing of the ADS
RFsM - an RFs identifying a "metabolic" status of the ADS
RFsE - an RFs identifying an "emotion" of "feeling" of the ADS
BFs - a set of RFs's identifying the doings or metabolism or feelings of the self ADS
RFM - a unique marker referring to an RF
BFM - a unique marker referring to a BF
PFM - an arbitrary unique marker to be used to refer to a PF
Rx - a "concrete" set of "sensed" RFM's referring to a sensed "concrete" object
Ri - an "abstract" set of RFM's
Px - a "concrete" set of allocated PFM's plus "concrete" RFM-set
Pi - a set of PFM's and RFM's referring to an "abstract object", often a single PFM
Ris - the "self" object
r - a specified spacio-temporal relationship, including the null relation
SE - a single-relation abstract situation, i.e., an r between among abstract objects, including a self object; thus SE = r(Ris,Pi1,Pi2,..)
Si - a Boolean set of SE's defining an abstract situation
dsd - an "incremental" action decision for a self-object
Lx - a location of a (usually-concrete) object, i.e., an Rx/Px
Vx - a vector of a (usually-concrete) object, i.e., an Rx/Px
Ex - an "event", for example described as a data row: Rx/Px...Lx...Vx...BF
Sx - a concrete situation described as a set of Ex's
PF - a "hierachied" rule that if (from the viewpoint of the tested object) the tested concrete situation (Sx) is included in the abstract situation (Si) of the rule, then DO the incremental action decision of the rule (dsd), each set of hierarchied PF's associated with an object being adequate for incremental simulation
SxC - the "now" Sx of an ADS, including access to the PF's referred to by the PFM's of the Px's of the Ex's
SxTn - (SxT1=SxC) from n=2, the nth incremental computation (in simulation) of the predicted situation of an ADS
Sip - an abstract situation (Si) of the self ADS which identifies a (hierarchied) problem p

FIG. 1B-2

Sigp - an abstract situation (Si) of the self ADS which identifies a (hierarchied) subgoal g attached to the problem p

dD - a hierarchied default real-time action decision attached to an abstract situation of the self ADS

dTd - a hierarchied trial (to try to get to a later subgoal without incurring a worse problem) incremental action decision attached to an abstract situation of the self ADS

dc - a "coordinated" real-time action decision for the ADS equal to (based on) a "successful" dTd (trial incremental decision)

G - a plan (a hierarchied set of subgoals g), each subgoal g having associated hierarchied dD's and dTd's

Gpn - the nth subgoal of a plan to solve the problem p

Pil - one of the "look-for" Pi's (abstract objects) of the next Sigp (of the next subgoal g), i.e., the next Gpn

RFl - one of the "look-for" RF's generated by finding (from a list of known concrete objects with their associated RF's) concrete objects including a current Pil and heuristically-selecting RF's to look for

FIG. 2

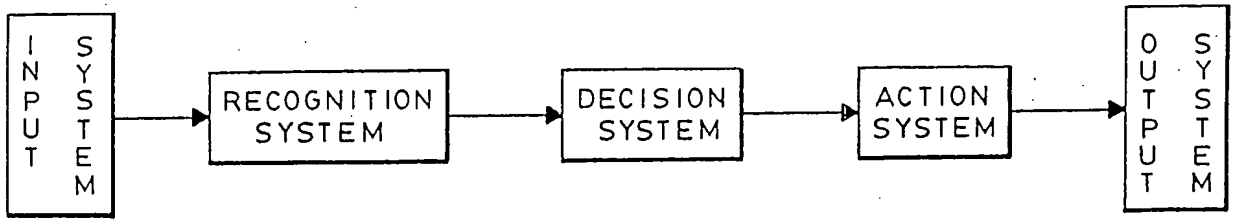


FIG. 3

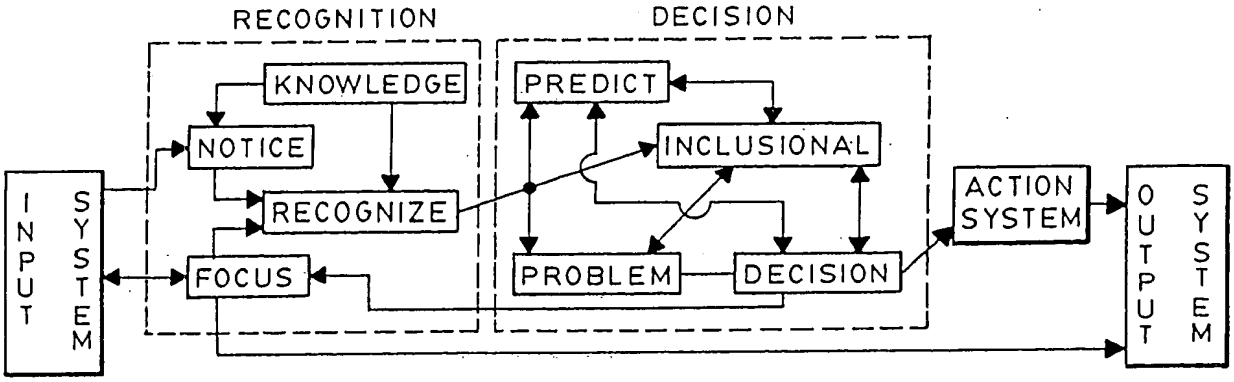


FIG. 4

DATA FLOW IN AUTONOMOUS DECISION SYSTEM

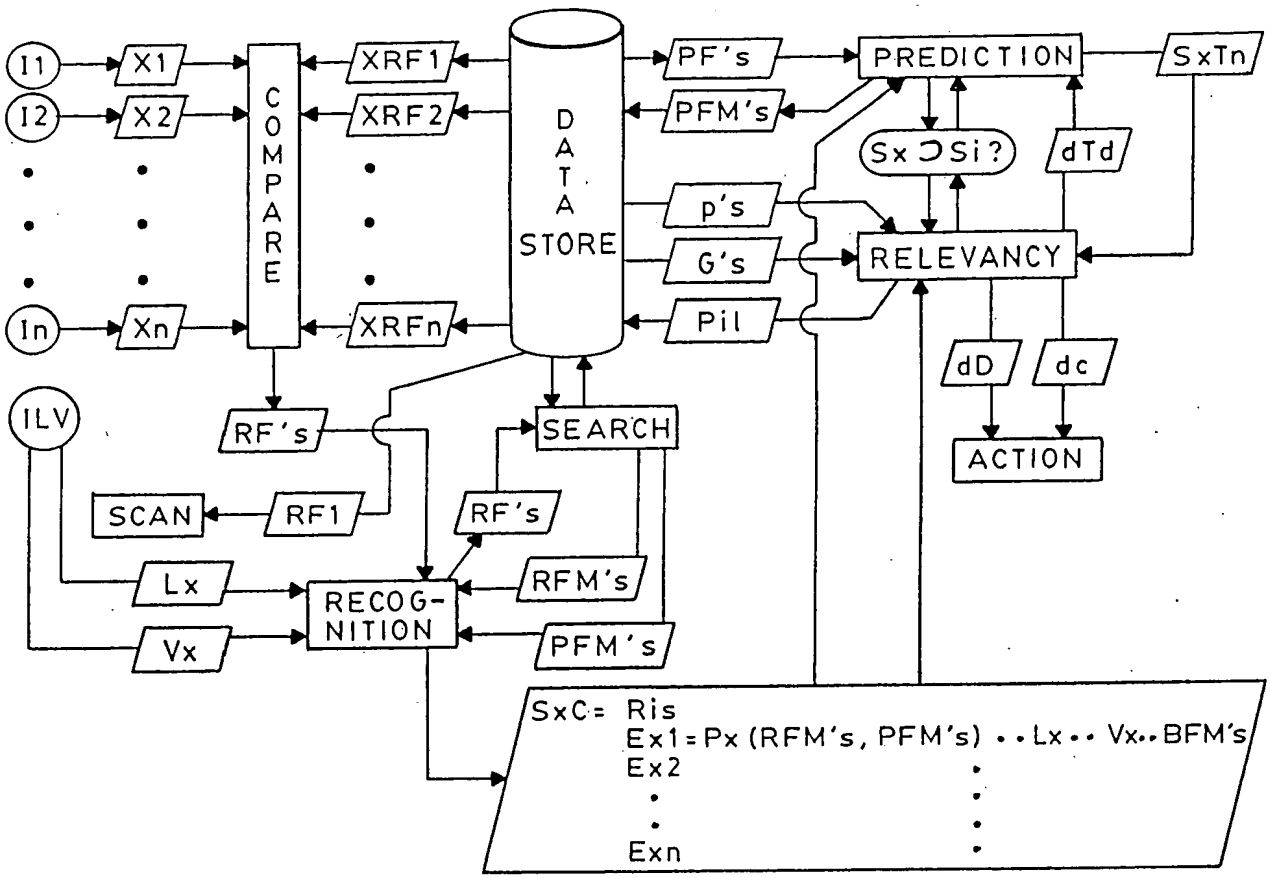
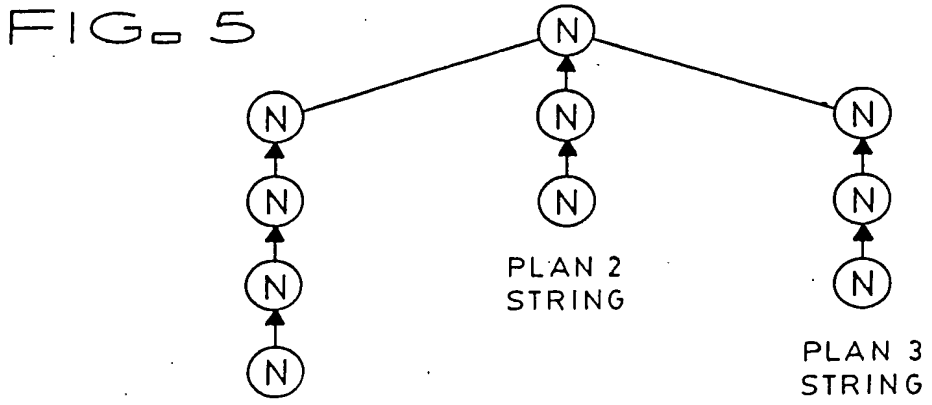


FIG. 5



- "N" MEANS A "NODE"
- UPPER-MOST NODE IS THE ABSENCE OF THE RELEVANT PROBLEM

FIG. 6

EXAMPLE COMPUTATION FLOW FOR TESTING WHETHER A GIVEN CONCRETE SITUATION (Sx) IS INCLUDED IN A GIVEN ABSTRACT SITUATION (Si), ONE SUB-EVALUATION (SE) AT A TIME:

Definition Reminders:

$$
Sx = \begin{pmatrix}
(Px1=[PFM_1, PFM_2, \ldots PFM_n; RFM_1, RFM_2 \ldots RFM_n] \ldots Lx_1 \ldots Vx_1 \ldots BFM's_1) \\
(Px2=[PFM_1, PFM_2, \ldots PFM_n; RFM_1, RFM_2 \ldots RFM_n] \ldots Lx_2 \ldots Vx_2 \ldots BFM's_2) \\
\vdots \\
(Pxn=[PFM_1, PFM_2, \ldots PFM_n; RFM_1, RFM_2 \ldots RFM_n] \ldots Lx_n \ldots Vx_n \ldots BFM's_n)
\end{pmatrix}
$$

SE= r(Pi1,Pi2,...Pin) — where any abstract thing (Pi) may specify one or more doings (BFM's) and where each Pi is a set of PFM's and a set of RFM's

----

Computation Flow:

IS Px1 INCLUDED IN Pi1? (i.e., IS Pi1 A SUBSET OF Px1? BFM'S?	IF YES, Px1 MAY STAND FOR Pi1 IN THE RELATION r (PROVIDED THE BFM'S OF Pi1, IF ANY, ARE A SUBSET OF THE BFM'S OF Px1)
IS Pi2 A SUBSET OF Px1? BFM'S?	IF YES, Px1 MAY STAND FOR Pi2
.	.
IS Pin A SUBSET OF Px1? BFM'S?	IF YES, Px1 MAY STAND FOR Pin
IS Pi1 A SUBSET OF Px2? BFM'S?	IF YES, Px2 MAY STAND FOR Pi1
.	.
IS Pin A SUBSET OF Px2? BFM'S?	IF YES, Px2 MAY STAND FOR Pin
.	.
IS Pi1 A SUBSET OF Pxn? BFM'S?	IF YES, Pxn MAY STAND FOR Pi1
.	.
IS Pin A SUBSET OF Pxn? BFM'S?	IF YES, Pxn MAY STAND FOR Pin

FOR ALL PERMUTATIONS OF Px'S STANDING FOR Pi'S, RESPECTIVELY, COMPUTE WHETHER THE r IS TRUE (FROM THE VIEWPOINT OF THE Lx/Vx OF THE APPROPRIATE "SELF" ROW OF THE Sx, I.E., THE APPROPRIATE Px) CONSIDERING THE Vx'S AND Lx'S OF OTHER Px'S STANDING FOR THE Pi'S OF THE RELATION r.

FOR ALL "TRUE" SUCH RELATION COMPUTATIONS, THERE IS SE INCLUSION. OTHERWISE, NOT. [FOR Si INCLUSION DETERMINATION, THE BOOLEAN COMPUTATIONS AMONG THE SE'S OF THE Si MUST BE DONE.]

FIG. 7A-1

```
{                    } Unit Si_mod; {                                    }
{ Purpose: ═══════════════════════════ Owner: TFL ═══════════════════════
  This is the main brain code. It determines the inclusionals for all
  decision making.
 History: ──────────────────────────────────────────────────────────────
 2/25/91: Rick added support for Story telling data using lecture mode
          and added Dans extra parm in the relations call.
 04/10/91 - Changed method of testing for body relations. Now any Row
            can do a test for HHT's by specifying itself as the 1st
            ri, ri2=to rfs that refer to HHT regs, and relop=FEEL.
 05/13/91: Added code to continue scanning SE's, even if current SE
          fails when the NextOp is BOR.
 07/29/91: Added code to allow to handle "sortof" of invalid SE's.
          If your SE spec's an RI that cant be found, then it was
          assumed FALSE. Now, if the preceeding NextOp is BNOT, then
          that boolean will be applied, and the SE will become TRUE.

 Copyright (c) 1989-1991 Anthrobotics ═══════════════════════════════════}
```

Upcoming changes to SIMOD, because of TFL.

- Onehit will be replaced with Nhits. This allows for dsd averaging. (this can be up to 20, the size of the datalist buffer.)
- Relative row/relrownum need to be expanded in SxRow to n hits.

```
{                    } Interface {                                       }
 Uses NewTypes,Types,Lists,Relation,SiStuff,DatList,Sxstuff;
 Const  MaxSiHits : Integer = 1;       {Tells how many SIStatuss to allow}
{-------------------------------------------------------------------------}
 PROCEDURE Situation_Inclusion(Var SiListToTest  : DataList; {What Sis to test;
set by Rxs}
                               Var SisThatHit    : DataList; {Returns list of Sis
that hit}
                               Var TheSx         : ListPtr;
                               Var SxIndex       : SxIndexArray;
                                   DefaultRow    : SxRowPtr);
{-------------------------------------------------------------------------}
{                    } Implementation {                                  }
 Uses RxStuff1,RiStuff,Strategy;
 Const BreakSi : Word = 0;  {used for debugging only!}
 {-------------------------------------------------------------------------}
{                      Situation Methods                                  }
 PROCEDURE Situation_Inclusion(Var SiListToTest  : DataList; {What Sis to test;
set by Rxs}
                               Var SisThatHit    : DataList; {Returns list of Sis
that hit}
                               Var TheSx         : ListPtr;
                               Var SxIndex    : SxIndexArray;
                                   DefaultRow    : SxRowPtr);
{ Purpose: ═══════════════════════════════════════════════════════════════
  This is it. The main cheese, where men are men and sheep are scared.
  This is where all the brain inclusionals occur.
                               Be in awe, mortal.                         }
 Var  SxRowPtr1,
      SxRowPtr2       : SxRowPtr;
      CurrentRI       : Array[0..1] of RiPtr;
      ValidRow        : Boolean;
      ANotSE,
      SIFailed,                        {Goes TRUE once we know weve failed.}
      SEPassedDefault,
      SEPassed,                        {The boolean status of the SE's so far.}
      SIStatus        : Boolean;
      CurrentSi       : SiPtr;
      SiNumber        : Byte;
      SENum      : Integer;
      SxRowCount1,
      SxRowCount2     : Byte;
      PrevHitRxs      : Array [0..1,1..MaxSesPerSI] of SxRowPtr;
      PrevHitRows     : Array[ 0..1,1..MaxSesPerSi] of Integer;
```

FIG. 7A-2

```
{-------------------------------------------------------------}
PROCEDURE Check_And_Set_PrevUsed_Ris(Var TheRow: SxRowPtr;
                                     RiNum,
                                     RiCol,
                                     SxRowNum : Integer);

Var Col,Row : Integer;
Begin
     PrevHitRows[RiCol,SENum]:=SxRowNum;
     PrevHitRxs[RiCol,SENum]:=TheRow;
     For Row:=1 to SENum do
         For Col:=0 to 1 do
           If CurrentSI^.SiRec.Ses[Row].TheRis[Col]=RiNum then
           Begin
                TheRow   := PrevHitRxs[Col,Row];
                PrevHitRows[RiCol,SENum]:=PrevHitRows[Col,Row];
                PrevHitRxs [RiCol,SENum]:=TheRow;
                exit;
           end;
end; {check and set prevused ris}
{-------------------------------------------------------------}
FUNCTION SxRow_In_SubEval(RiNum,RiCol: Integer; Var TheSxRow: SxRowPtr; Var
Row: Byte): Boolean;
Begin
     SxRow_In_SubEval:=True;
     TheSxRow:=DefaultRow;
     If RiNum<>SelfRI then
        Begin
             TheSxRow := SxRowPtr(TheSx^.Find_Element(SxIndex[Row].IndexPos));
             If RiNum<>AnyRi then
                SxRow_In_SubEval:=TheSxRow^.Matches_Ri(CurrentRi[RiCol])
        end
     else If (Row>1) and (RiNum=SelfRi) then
          SxRow_In_SubEval:=False
end; {SxRow in subeval}
{-------------------------------------------------------------}
FUNCTION Find_Ri(RiNum,RiCol,SkipRow: Integer; Var SxRow: SxRowPtr; Var Row:
Byte): Boolean;
Var Hit : Boolean;
Begin
     Hit:=False;
     If RiNum<>AnyRi then
        While (Not Hit) and (Row<TheSx^.Count) do
        Begin
             Inc(Row);
             If Row<>SkipRow then
                Begin
                     Hit:=SxRow_in_SubEval(RiNum,RiCol,SxRow,Row);
                     If Hit then
                        Check_And_Set_PrevUsed_Ris(SxRow,RiNum,RiCol,Row)
                     else SxRow:=Nil
                end
             else SxRow:=Nil;
        end
     else
        Begin
             Hit:=True;
             Row:=TheSx^.Count;
             SxRow:=SxRowPtr(TheSx^.Find_Element(Row));
        end;
     Find_Ri:=Hit;
end; {find ri}
{-------------------------------------------------------------}
PROCEDURE Combine_Boolean;
Begin
     With CurrentSI^.SIRec do
     Begin
          If SENum>1 then with Ses[SENum-1] do
               Case NextOp of
                    BNot,
                    BAnd : SIStatus := SIStatus and SEPassed;
                    BXor : SIStatus := SIStatus Xor SEPassed;
```

FIG. 7A-3

```
                    BOr  : SIStatus := SIStatus or  SEPassed;
                end
            else SIStatus:=SEPassed;
            If (SEPassed) then DefaultRow^.Set_Rel_Row(SxRowPtr2);
            SIFailed:=(SIStatus=False) and (Ses[SENum].NextOp=BAnd); {lookahead
boolean shortcut}
        end;
  end; {Combine boolean}
{-------------------------------------------------------------}
 Begin
     For SiNumber:=1 to SiListToTest.Drec.Used do
         If (SiListToTest.Drec.Data[SINumber]<>0) then {loop for each SI until you
have a hit...}
     Begin
          If SiListToTest.Drec.Data[SiNumber]=BreakSI then {DEBUG}
             SENum:=0;
          CurrentSI :=
SiPtr(SiList.Find_Element(SiListToTest.DRec.Data[SiNumber]));  {Get current SI}
          SIStatus  := True;  {Until proven guilty...}
          SIFailed  := False;
          SENum     := 0;
          FillChar( PrevHitRxs,  SizeOf(PrevHitRxs), 0 );  {Clear PrevHitRxs}
          FillChar( PrevHitRows, SizeOf(PrevHitRxs), 0 );
          With CurrentSI^.SIRec do
          Repeat                          {Until no more subevals, or SI fails.}
               Inc(SENum);
               SxRowCount1:=0;
               ANotSE:=(SENum>1) and                  {This handles special case
where se is true if}
                    (SEs[SENum-1].NextOp=BNot);  {no 2 RIs hit, but nextop =
NOT; (SE true)}
               SEPassedDefault:=ANotSE;
               SEPassed:=False;
               CurrentRi[0]:=RiPtr(RiList.Find_Element(Ses[SENum].TheRis[0]));
               With Ses[SENum] do
                 Repeat {find ri0 loop}
                     If Find_RI(TheRis[0],0,0,SxRowPtr1,SxRowCount1) then
                       Begin
                          SxRowCount2:=0;  {search from top of sxlist}
                          Repeat {find ri1 loop}
                            CurrentRi[1] =RiPtr(RiList.Find_Element(TheRis[1]));
                            If Op=IsFeeling then
                             Begin
                               SxRowCount2:=TheSx^.Count;
                               SxRowPtr2:=SxRowPtr1;
                               SEPassed:=(SxRowPtr1^.Matches_Ri(CurrentRi[1]));
                               If ANotSE then
                                 SEPassed:= (Not SEPassed);
                               SEPassedDefault:=SEPassed;
                             end
                            else
                              If
Find_RI(TheRis[1],1,SxRowCount1,SxRowPtr2,SxRowCount2) then
                                Begin
                                  SEPassed:=(Relations[Op](SxRowPtr1^,SxRowPtr2^,
DefaultRow^,OpParm));
                                  If ANotSE then SEPassed:=Not SEPassed;
                                  SEPassedDefault:=SEPassed;
                                end
                             else If SxRowCount2>=TheSx^.Count then
                               SEPassed:=SEPassedDefault;
                          Until (SEPassed) or (SxRowCount2>=TheSx^.Count);
                       end
                     else SEPassed:=SEPassedDefault;
                 Until (SxRowCount1>=TheSx^.Count) or (SEPassed);
               Combine_Boolean;
          Until (SIFailed) or                  {means this SI has definately failed.}
          (Ses[SENum].NextOp=NoOp);  {we hit last SE in the Si}
        If (SIStatus) and
        (SisThatHit.AppendItem(1,SiNumber)) then
          exit;
     end;
 end; {Situation inclusional}{si mod unit}
```

FIG. 8

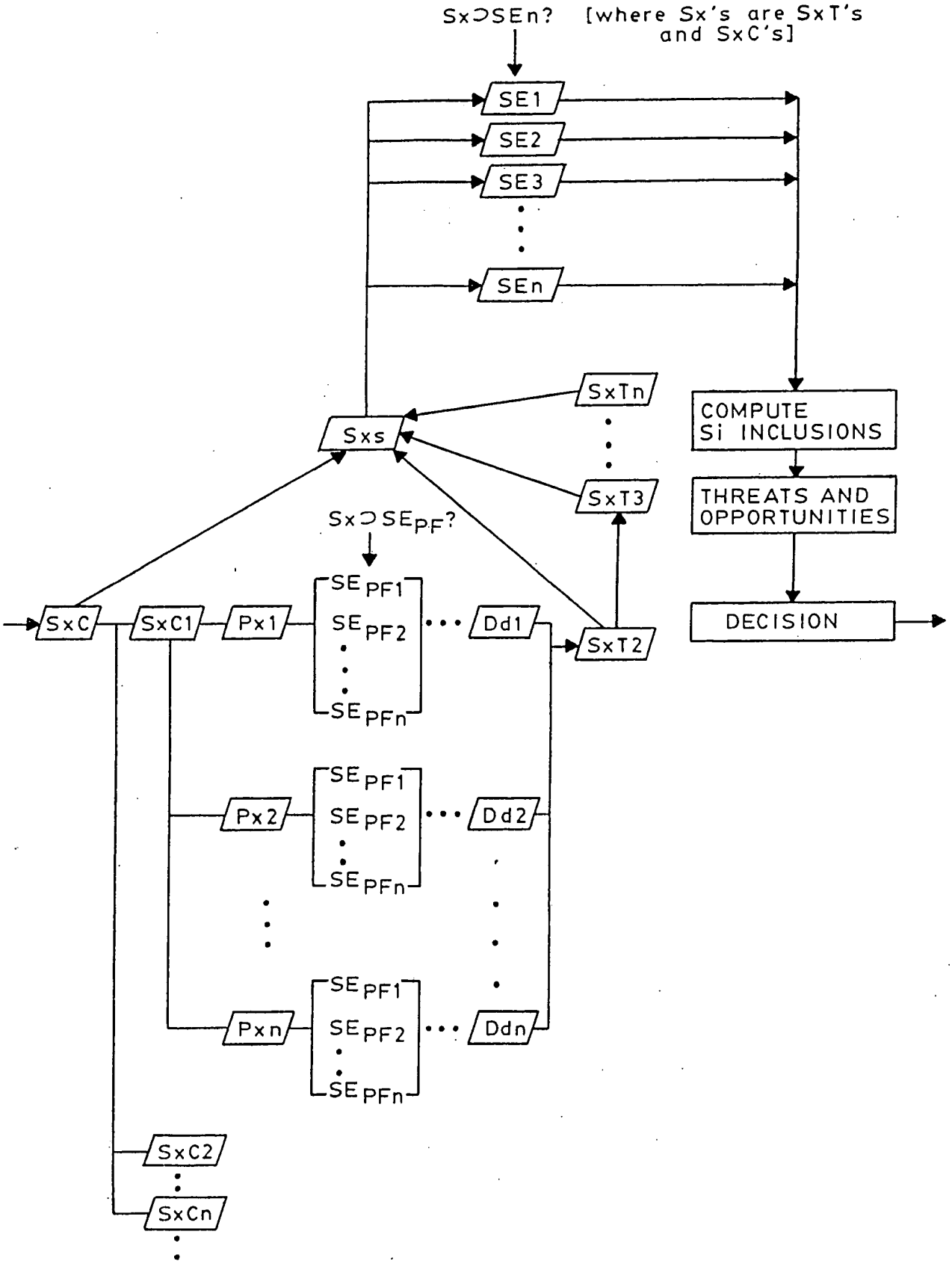

SxSEn? [where Sx's are SxT's and SxC's]
SE1
SE2
SE3
SEn
SxTn
Sxs
SxT3
COMPUTE Si INCLUSIONS
THREATS AND OPPORTUNITIES
DECISION
SxSEPF?
SxC
SxC1
Px1
SEPF1
SEPF2
SEPFn
Dd1
SxT2
Px2
Dd2
Pxn
Ddn
SxC2
SxCn

FIG. 9A-1

STEPS IN A COMPUTER PROGRAM IMPLEMENTING NATURAL LANGUAGE AND STORY PRODUCTION IN AUTONOMOUS DECISION SYSTEMS

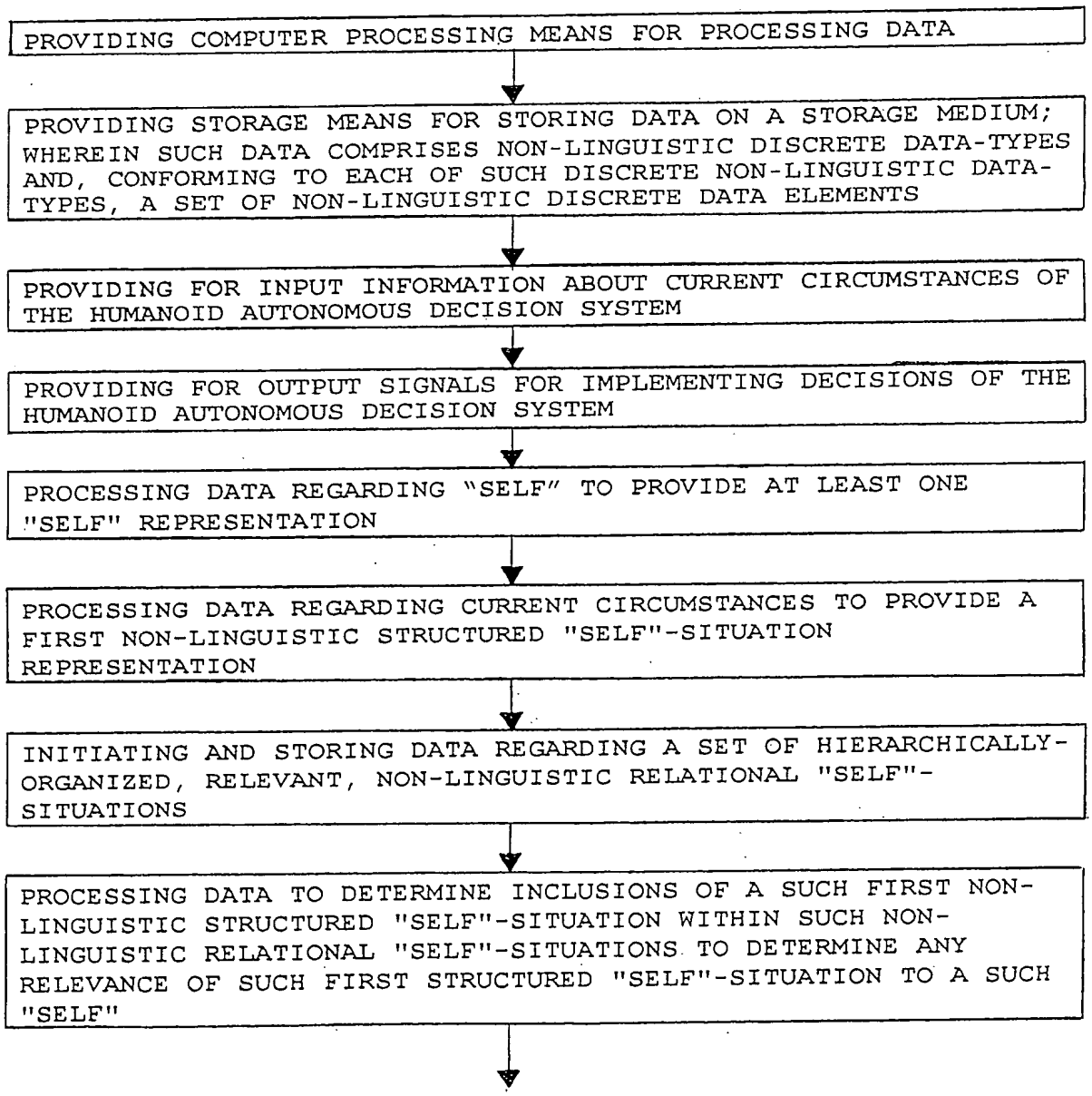


FIG. 9A-2

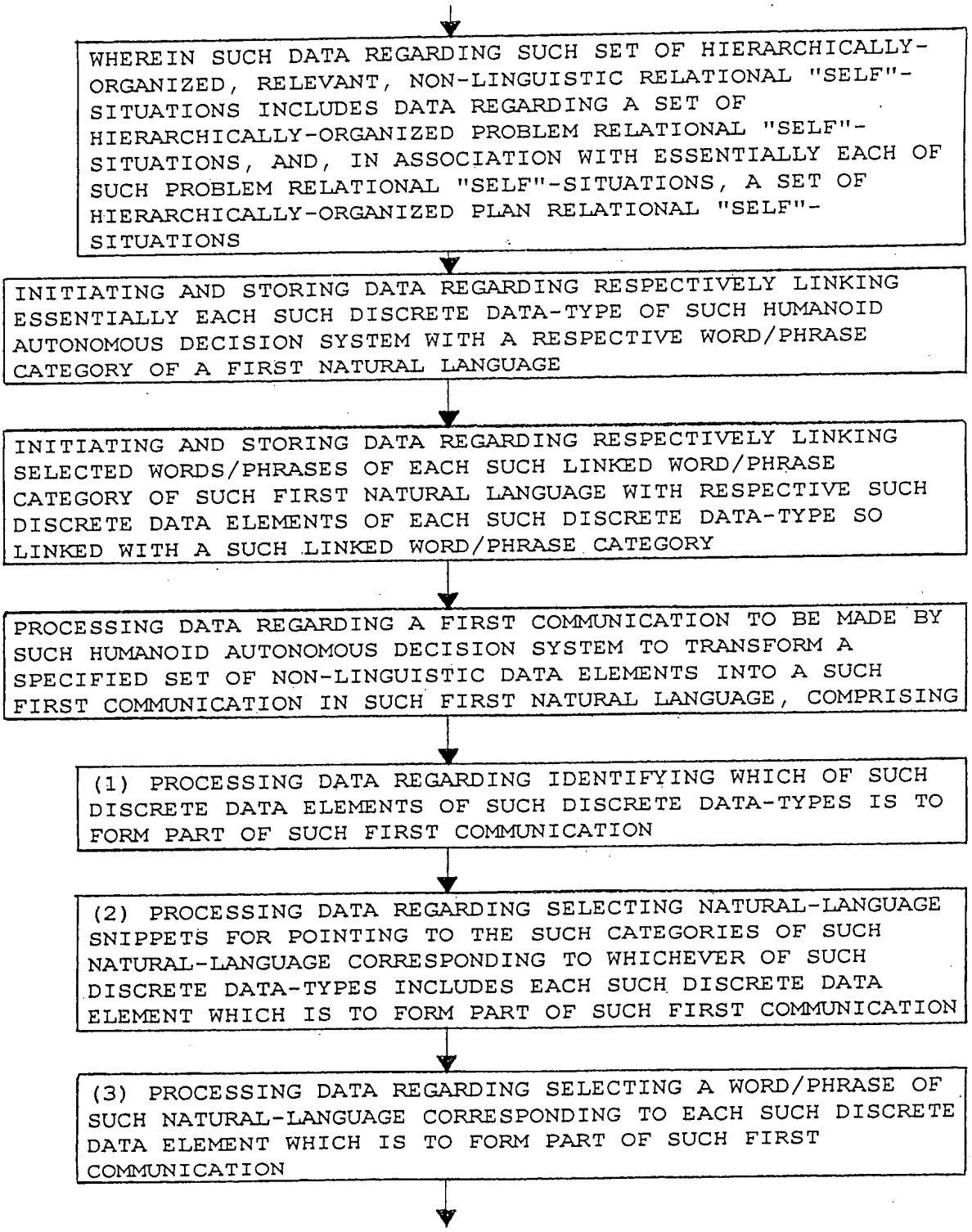

WHEREIN SUCH DATA REGARDING SUCH SET OF HIERARCHICALLY-ORGANIZED, RELEVANT, NON-LINGUISTIC RELATIONAL "SELF"-SITUATIONS INCLUDES DATA REGARDING A SET OF HIERARCHICALLY-ORGANIZED PROBLEM RELATIONAL "SELF"-SITUATIONS, AND, IN ASSOCIATION WITH ESSENTIALLY EACH OF SUCH PROBLEM RELATIONAL "SELF"-SITUATIONS, A SET OF HIERARCHICALLY-ORGANIZED PLAN RELATIONAL "SELF"-SITUATIONS
INITIATING AND STORING DATA REGARDING RESPECTIVELY LINKING ESSENTIALLY EACH SUCH DISCRETE DATA-TYPE OF SUCH HUMANOID AUTONOMOUS DECISION SYSTEM WITH A RESPECTIVE WORD/PHRASE CATEGORY OF A FIRST NATURAL LANGUAGE
INITIATING AND STORING DATA REGARDING RESPECTIVELY LINKING SELECTED WORDS/PHRASES OF EACH SUCH LINKED WORD/PHRASE CATEGORY OF SUCH FIRST NATURAL LANGUAGE WITH RESPECTIVE SUCH DISCRETE DATA ELEMENTS OF EACH SUCH DISCRETE DATA-TYPE SO LINKED WITH A SUCH LINKED WORD/PHRASE CATEGORY
PROCESSING DATA REGARDING A FIRST COMMUNICATION TO BE MADE BY SUCH HUMANOID AUTONOMOUS DECISION SYSTEM TO TRANSFORM A SPECIFIED SET OF NON-LINGUISTIC DATA ELEMENTS INTO A SUCH FIRST COMMUNICATION IN SUCH FIRST NATURAL LANGUAGE, COMPRISING
(1) PROCESSING DATA REGARDING IDENTIFYING WHICH OF SUCH DISCRETE DATA ELEMENTS OF SUCH DISCRETE DATA-TYPES IS TO FORM PART OF SUCH FIRST COMMUNICATION
(2) PROCESSING DATA REGARDING SELECTING NATURAL-LANGUAGE SNIPPETS FOR POINTING TO THE SUCH CATEGORIES OF SUCH NATURAL-LANGUAGE CORRESPONDING TO WHICHEVER OF SUCH DISCRETE DATA-TYPES INCLUDES EACH SUCH DISCRETE DATA ELEMENT WHICH IS TO FORM PART OF SUCH FIRST COMMUNICATION
(3) PROCESSING DATA REGARDING SELECTING A WORD/PHRASE OF SUCH NATURAL-LANGUAGE CORRESPONDING TO EACH SUCH DISCRETE DATA ELEMENT WHICH IS TO FORM PART OF SUCH FIRST COMMUNICATION

FIG. 9A-3

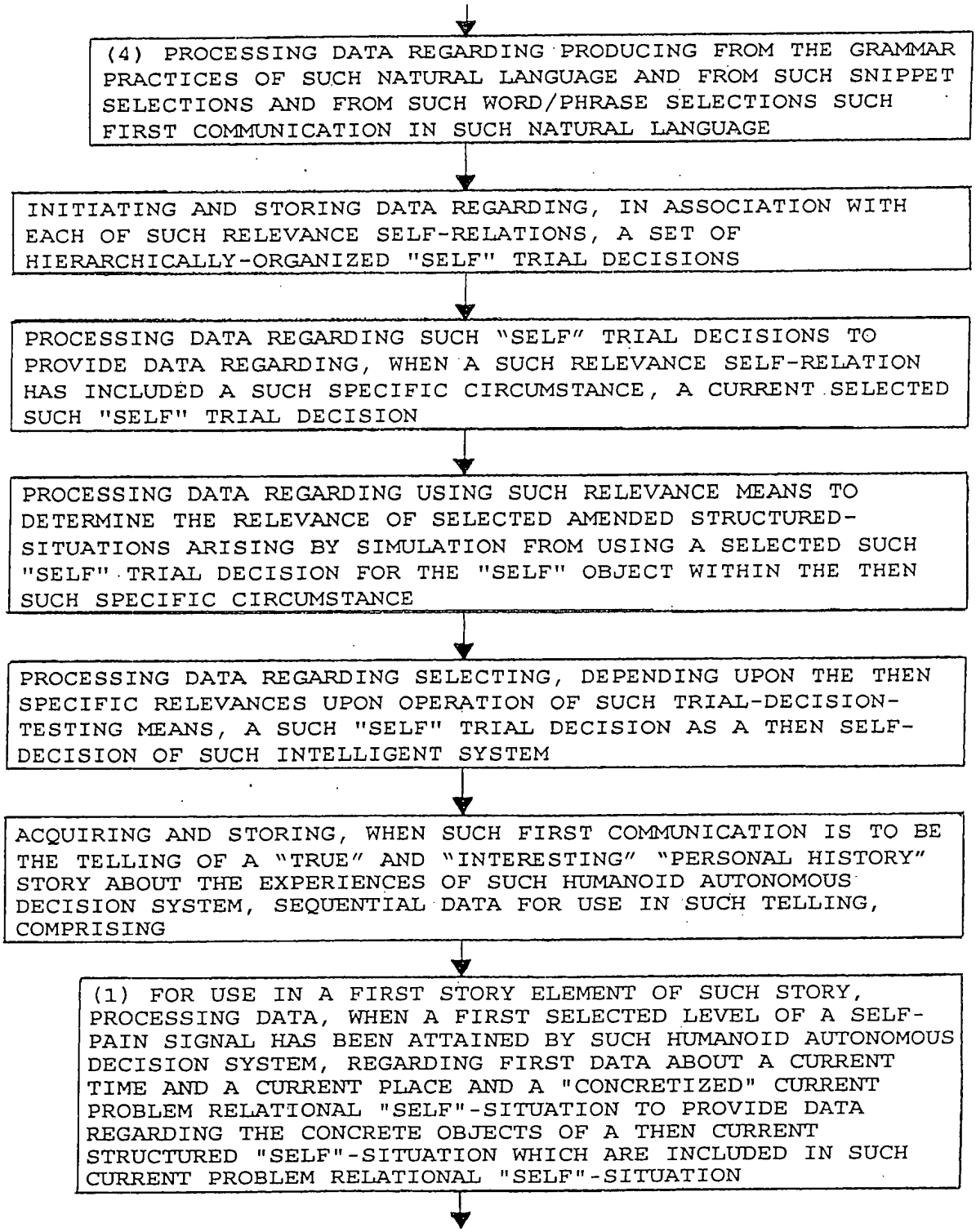

(4) PROCESSING DATA REGARDING PRODUCING FROM THE GRAMMAR PRACTICES OF SUCH NATURAL LANGUAGE AND FROM SUCH SNIPPET SELECTIONS AND FROM SUCH WORD/PHRASE SELECTIONS SUCH FIRST COMMUNICATION IN SUCH NATURAL LANGUAGE
INITIATING AND STORING DATA REGARDING, IN ASSOCIATION WITH EACH OF SUCH RELEVANCE SELF-RELATIONS, A SET OF HIERARCHICALLY-ORGANIZED "SELF" TRIAL DECISIONS
PROCESSING DATA REGARDING SUCH "SELF" TRIAL DECISIONS TO PROVIDE DATA REGARDING, WHEN A SUCH RELEVANCE SELF-RELATION HAS INCLUDED A SUCH SPECIFIC CIRCUMSTANCE, A CURRENT SELECTED SUCH "SELF" TRIAL DECISION
PROCESSING DATA REGARDING USING SUCH RELEVANCE MEANS TO DETERMINE THE RELEVANCE OF SELECTED AMENDED STRUCTURED-SITUATIONS ARISING BY SIMULATION FROM USING A SELECTED SUCH "SELF" TRIAL DECISION FOR THE "SELF" OBJECT WITHIN THE THEN SUCH SPECIFIC CIRCUMSTANCE
PROCESSING DATA REGARDING SELECTING, DEPENDING UPON THE THEN SPECIFIC RELEVANCES UPON OPERATION OF SUCH TRIAL-DECISION-TESTING MEANS, A SUCH "SELF" TRIAL DECISION AS A THEN SELF-DECISION OF SUCH INTELLIGENT SYSTEM
ACQUIRING AND STORING, WHEN SUCH FIRST COMMUNICATION IS TO BE THE TELLING OF A "TRUE" AND "INTERESTING" "PERSONAL HISTORY" STORY ABOUT THE EXPERIENCES OF SUCH HUMANOID AUTONOMOUS DECISION SYSTEM, SEQUENTIAL DATA FOR USE IN SUCH TELLING, COMPRISING
(1) FOR USE IN A FIRST STORY ELEMENT OF SUCH STORY, PROCESSING DATA, WHEN A FIRST SELECTED LEVEL OF A SELF-PAIN SIGNAL HAS BEEN ATTAINED BY SUCH HUMANOID AUTONOMOUS DECISION SYSTEM, REGARDING FIRST DATA ABOUT A CURRENT TIME AND A CURRENT PLACE AND A "CONCRETIZED" CURRENT PROBLEM RELATIONAL "SELF"-SITUATION TO PROVIDE DATA REGARDING THE CONCRETE OBJECTS OF A THEN CURRENT STRUCTURED "SELF"-SITUATION WHICH ARE INCLUDED IN SUCH CURRENT PROBLEM RELATIONAL "SELF"-SITUATION

FIG. 9A-4

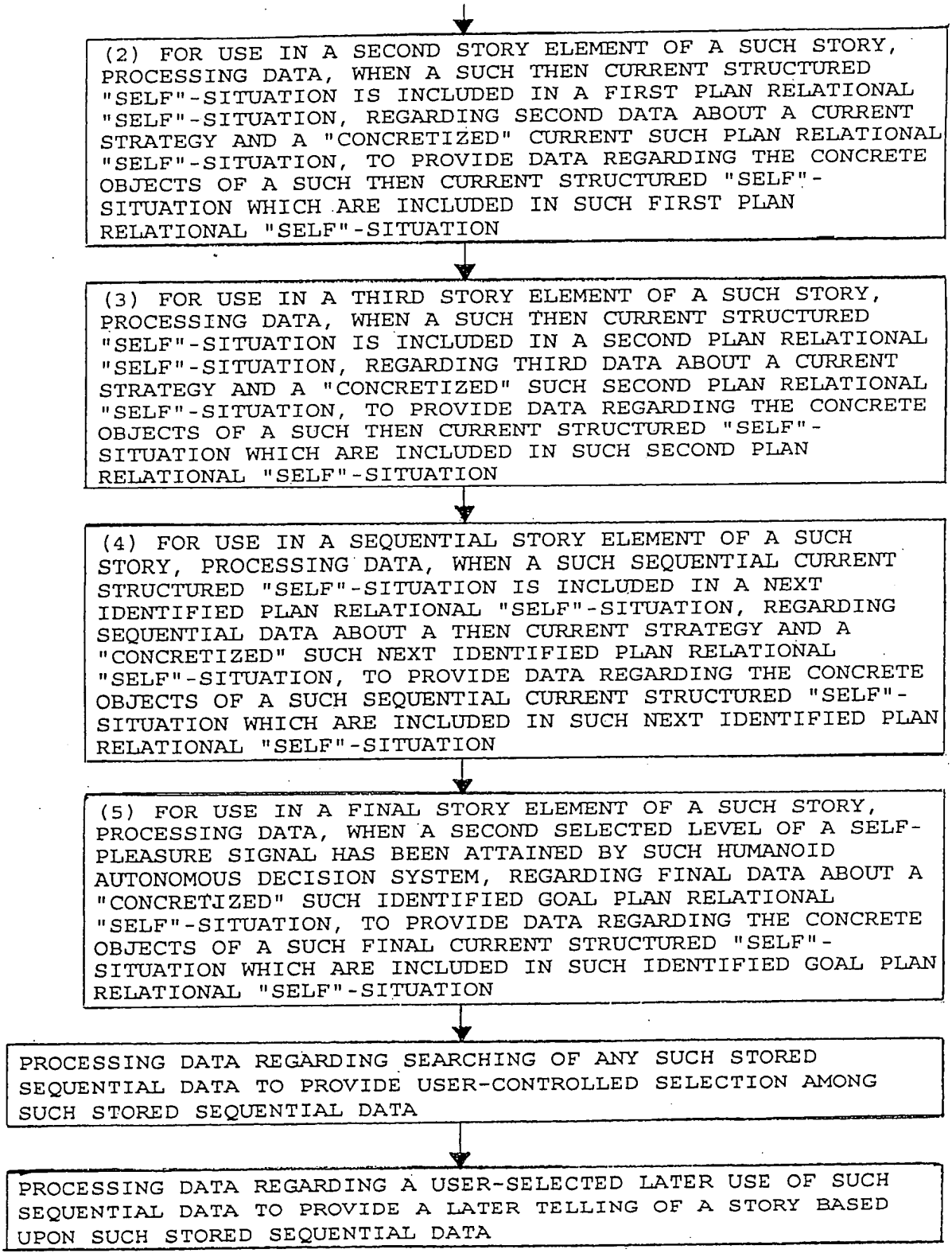

(2) FOR USE IN A SECOND STORY ELEMENT OF A SUCH STORY, PROCESSING DATA, WHEN A SUCH THEN CURRENT STRUCTURED "SELF"-SITUATION IS INCLUDED IN A FIRST PLAN RELATIONAL "SELF"-SITUATION, REGARDING SECOND DATA ABOUT A CURRENT STRATEGY AND A "CONCRETIZED" CURRENT SUCH PLAN RELATIONAL "SELF"-SITUATION, TO PROVIDE DATA REGARDING THE CONCRETE OBJECTS OF A SUCH THEN CURRENT STRUCTURED "SELF"-SITUATION WHICH ARE INCLUDED IN SUCH FIRST PLAN RELATIONAL "SELF"-SITUATION
(3) FOR USE IN A THIRD STORY ELEMENT OF A SUCH STORY, PROCESSING DATA, WHEN A SUCH THEN CURRENT STRUCTURED "SELF"-SITUATION IS INCLUDED IN A SECOND PLAN RELATIONAL "SELF"-SITUATION, REGARDING THIRD DATA ABOUT A CURRENT STRATEGY AND A "CONCRETIZED" SUCH SECOND PLAN RELATIONAL "SELF"-SITUATION, TO PROVIDE DATA REGARDING THE CONCRETE OBJECTS OF A SUCH THEN CURRENT STRUCTURED "SELF"-SITUATION WHICH ARE INCLUDED IN SUCH SECOND PLAN RELATIONAL "SELF"-SITUATION
(4) FOR USE IN A SEQUENTIAL STORY ELEMENT OF A SUCH STORY, PROCESSING DATA, WHEN A SUCH SEQUENTIAL CURRENT STRUCTURED "SELF"-SITUATION IS INCLUDED IN A NEXT IDENTIFIED PLAN RELATIONAL "SELF"-SITUATION, REGARDING SEQUENTIAL DATA ABOUT A THEN CURRENT STRATEGY AND A "CONCRETIZED" SUCH NEXT IDENTIFIED PLAN RELATIONAL "SELF"-SITUATION, TO PROVIDE DATA REGARDING THE CONCRETE OBJECTS OF A SUCH SEQUENTIAL CURRENT STRUCTURED "SELF"-SITUATION WHICH ARE INCLUDED IN SUCH NEXT IDENTIFIED PLAN RELATIONAL "SELF"-SITUATION
(5) FOR USE IN A FINAL STORY ELEMENT OF A SUCH STORY, PROCESSING DATA, WHEN A SECOND SELECTED LEVEL OF A SELF-PLEASURE SIGNAL HAS BEEN ATTAINED BY SUCH HUMANOID AUTONOMOUS DECISION SYSTEM, REGARDING FINAL DATA ABOUT A "CONCRETIZED" SUCH IDENTIFIED GOAL PLAN RELATIONAL "SELF"-SITUATION, TO PROVIDE DATA REGARDING THE CONCRETE OBJECTS OF A SUCH FINAL CURRENT STRUCTURED "SELF"-SITUATION WHICH ARE INCLUDED IN SUCH IDENTIFIED GOAL PLAN RELATIONAL "SELF"-SITUATION
PROCESSING DATA REGARDING SEARCHING OF ANY SUCH STORED SEQUENTIAL DATA TO PROVIDE USER-CONTROLLED SELECTION AMONG SUCH STORED SEQUENTIAL DATA
PROCESSING DATA REGARDING A USER-SELECTED LATER USE OF SUCH SEQUENTIAL DATA TO PROVIDE A LATER TELLING OF A STORY BASED UPON SUCH STORED SEQUENTIAL DATA

FIG. 9B-1

STEPS IN A COMPUTER PROGRAM FOR IMPLEMENTING FIRST NATURAL LANGUAGE INTERPRETATION FUNCTIONS IN A HUMANOID AUTONOMOUS DECISION SYSTEM INTERPRETING INCOMING FIRST NATURAL LANGUAGE FROM AN OTHER

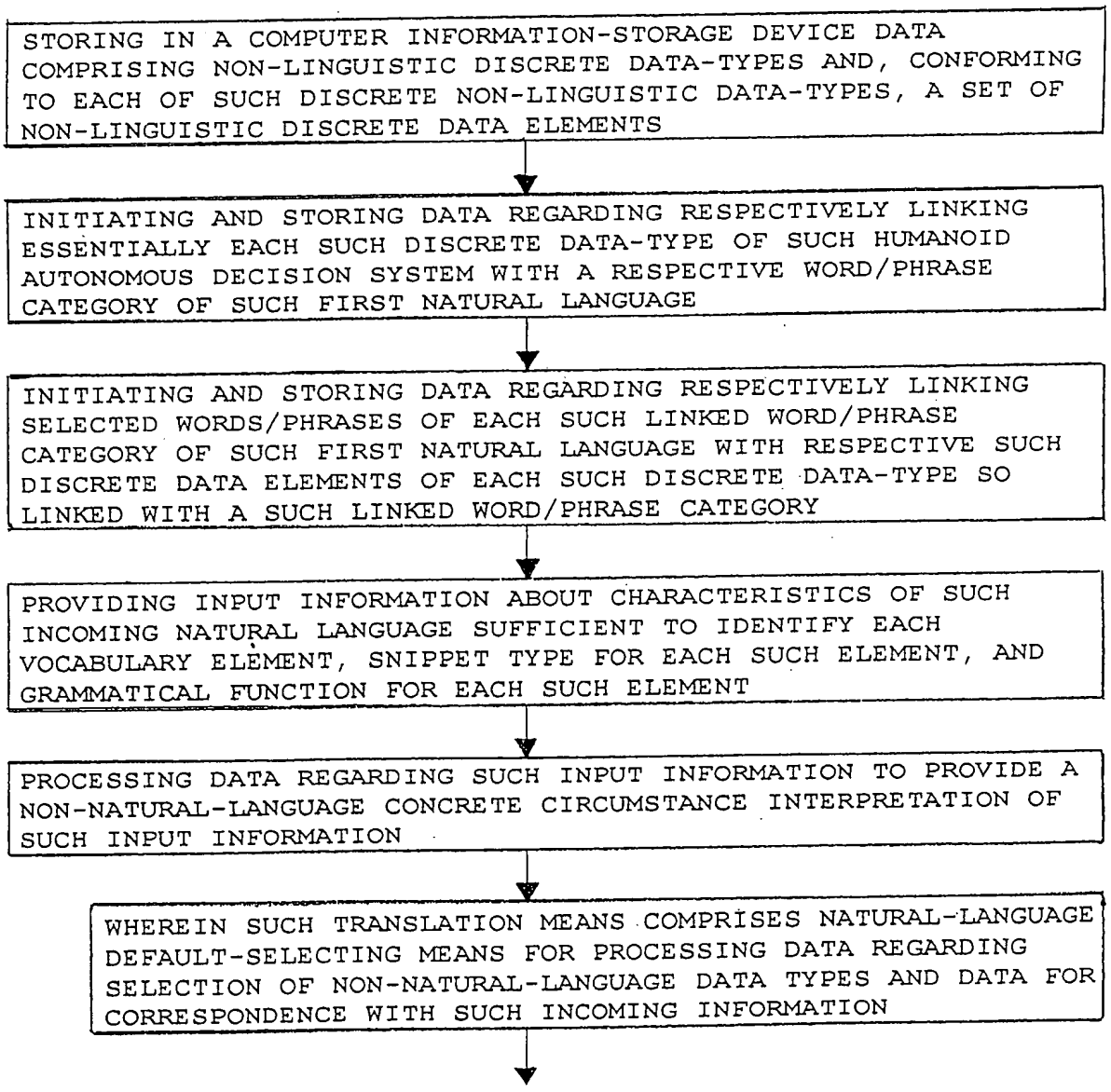


FIG. 9B-2

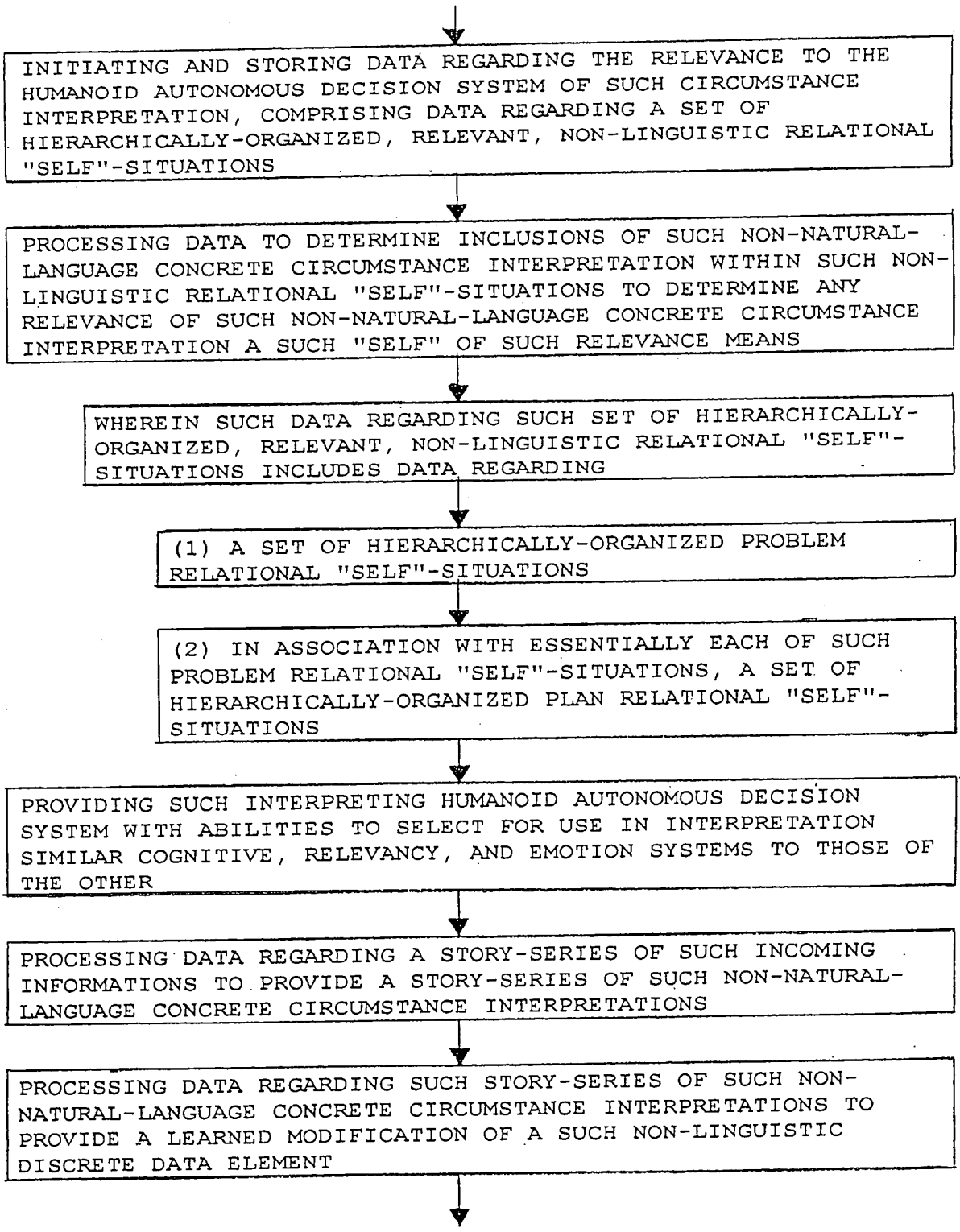

INITIATING AND STORING DATA REGARDING THE RELEVANCE TO THE HUMANOID AUTONOMOUS DECISION SYSTEM OF SUCH CIRCUMSTANCE INTERPRETATION, COMPRISING DATA REGARDING A SET OF HIERARCHICALLY-ORGANIZED, RELEVANT, NON-LINGUISTIC RELATIONAL "SELF"-SITUATIONS
PROCESSING DATA TO DETERMINE INCLUSIONS OF SUCH NON-NATURAL-LANGUAGE CONCRETE CIRCUMSTANCE INTERPRETATION WITHIN SUCH NON-LINGUISTIC RELATIONAL "SELF"-SITUATIONS TO DETERMINE ANY RELEVANCE OF SUCH NON-NATURAL-LANGUAGE CONCRETE CIRCUMSTANCE INTERPRETATION A SUCH "SELF" OF SUCH RELEVANCE MEANS
WHEREIN SUCH DATA REGARDING SUCH SET OF HIERARCHICALLY-ORGANIZED, RELEVANT, NON-LINGUISTIC RELATIONAL "SELF"-SITUATIONS INCLUDES DATA REGARDING
(1) A SET OF HIERARCHICALLY-ORGANIZED PROBLEM RELATIONAL "SELF"-SITUATIONS
(2) IN ASSOCIATION WITH ESSENTIALLY EACH OF SUCH PROBLEM RELATIONAL "SELF"-SITUATIONS, A SET OF HIERARCHICALLY-ORGANIZED PLAN RELATIONAL "SELF"-SITUATIONS
PROVIDING SUCH INTERPRETING HUMANOID AUTONOMOUS DECISION SYSTEM WITH ABILITIES TO SELECT FOR USE IN INTERPRETATION SIMILAR COGNITIVE, RELEVANCY, AND EMOTION SYSTEMS TO THOSE OF THE OTHER
PROCESSING DATA REGARDING A STORY-SERIES OF SUCH INCOMING INFORMATIONS TO PROVIDE A STORY-SERIES OF SUCH NON-NATURAL-LANGUAGE CONCRETE CIRCUMSTANCE INTERPRETATIONS
PROCESSING DATA REGARDING SUCH STORY-SERIES OF SUCH NON-NATURAL-LANGUAGE CONCRETE CIRCUMSTANCE INTERPRETATIONS TO PROVIDE A LEARNED MODIFICATION OF A SUCH NON-LINGUISTIC DISCRETE DATA ELEMENT

FIG. 9B-3

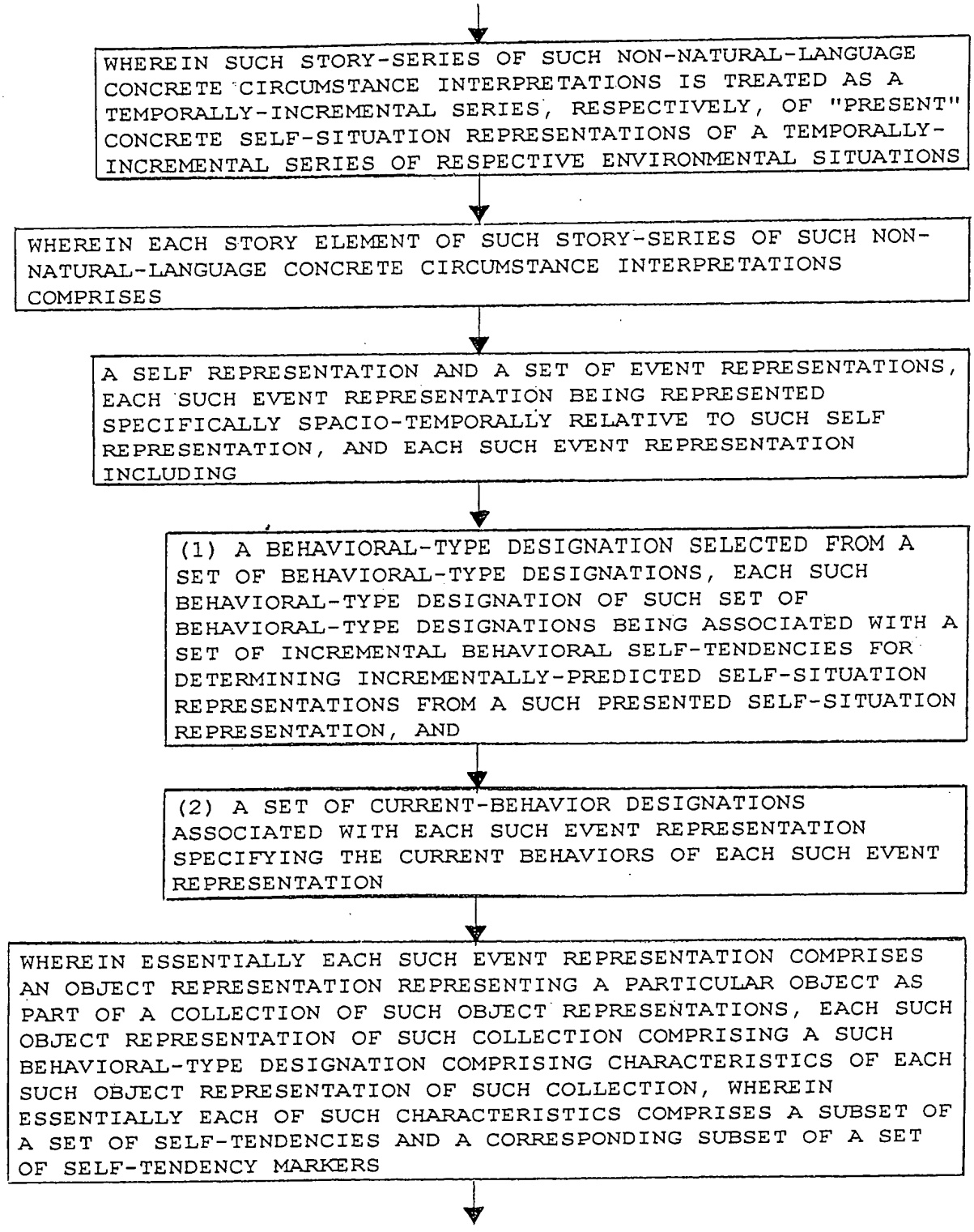

WHEREIN SUCH STORY-SERIES OF SUCH NON-NATURAL-LANGUAGE CONCRETE CIRCUMSTANCE INTERPRETATIONS IS TREATED AS A TEMPORALLY-INCREMENTAL SERIES, RESPECTIVELY, OF "PRESENT" CONCRETE SELF-SITUATION REPRESENTATIONS OF A TEMPORALLY-INCREMENTAL SERIES OF RESPECTIVE ENVIRONMENTAL SITUATIONS
WHEREIN EACH STORY ELEMENT OF SUCH STORY-SERIES OF SUCH NON-NATURAL-LANGUAGE CONCRETE CIRCUMSTANCE INTERPRETATIONS COMPRISES
A SELF REPRESENTATION AND A SET OF EVENT REPRESENTATIONS, EACH SUCH EVENT REPRESENTATION BEING REPRESENTED SPECIFICALLY SPACIO-TEMPORALLY RELATIVE TO SUCH SELF REPRESENTATION, AND EACH SUCH EVENT REPRESENTATION INCLUDING
(1) A BEHAVIORAL-TYPE DESIGNATION SELECTED FROM A SET OF BEHAVIORAL-TYPE DESIGNATIONS, EACH SUCH BEHAVIORAL-TYPE DESIGNATION OF SUCH SET OF BEHAVIORAL-TYPE DESIGNATIONS BEING ASSOCIATED WITH A SET OF INCREMENTAL BEHAVIORAL SELF-TENDENCIES FOR DETERMINING INCREMENTALLY-PREDICTED SELF-SITUATION REPRESENTATIONS FROM A SUCH PRESENTED SELF-SITUATION REPRESENTATION, AND
(2) A SET OF CURRENT-BEHAVIOR DESIGNATIONS ASSOCIATED WITH EACH SUCH EVENT REPRESENTATION SPECIFYING THE CURRENT BEHAVIORS OF EACH SUCH EVENT REPRESENTATION
WHEREIN ESSENTIALLY EACH SUCH EVENT REPRESENTATION COMPRISES AN OBJECT REPRESENTATION REPRESENTING A PARTICULAR OBJECT AS PART OF A COLLECTION OF SUCH OBJECT REPRESENTATIONS, EACH SUCH OBJECT REPRESENTATION OF SUCH COLLECTION COMPRISING A SUCH BEHAVIORAL-TYPE DESIGNATION COMPRISING CHARACTERISTICS OF EACH SUCH OBJECT REPRESENTATION OF SUCH COLLECTION, WHEREIN ESSENTIALLY EACH OF SUCH CHARACTERISTICS COMPRISES A SUBSET OF A SET OF SELF-TENDENCIES AND A CORRESPONDING SUBSET OF A SET OF SELF-TENDENCY MARKERS

FIG. 9B-4

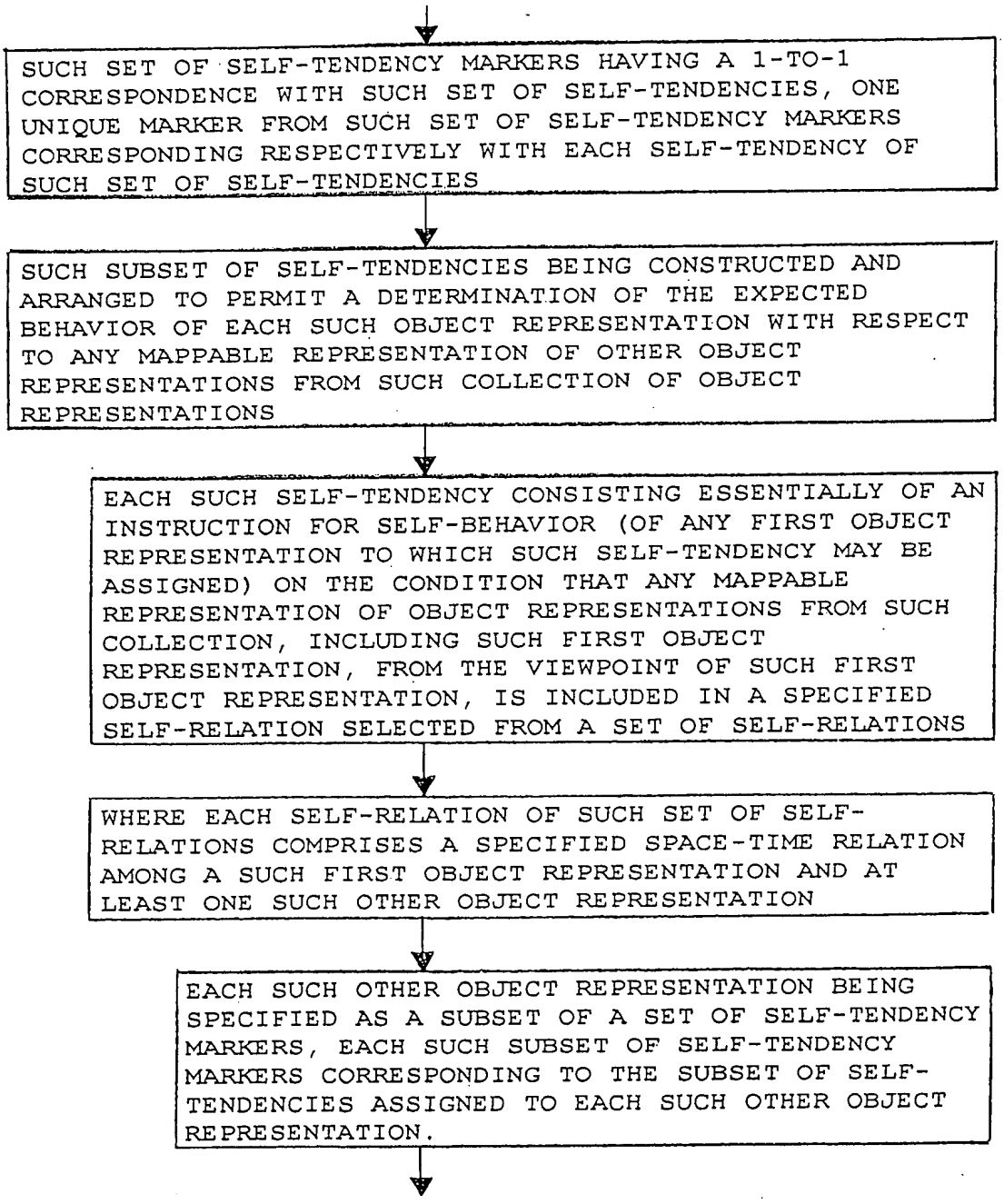
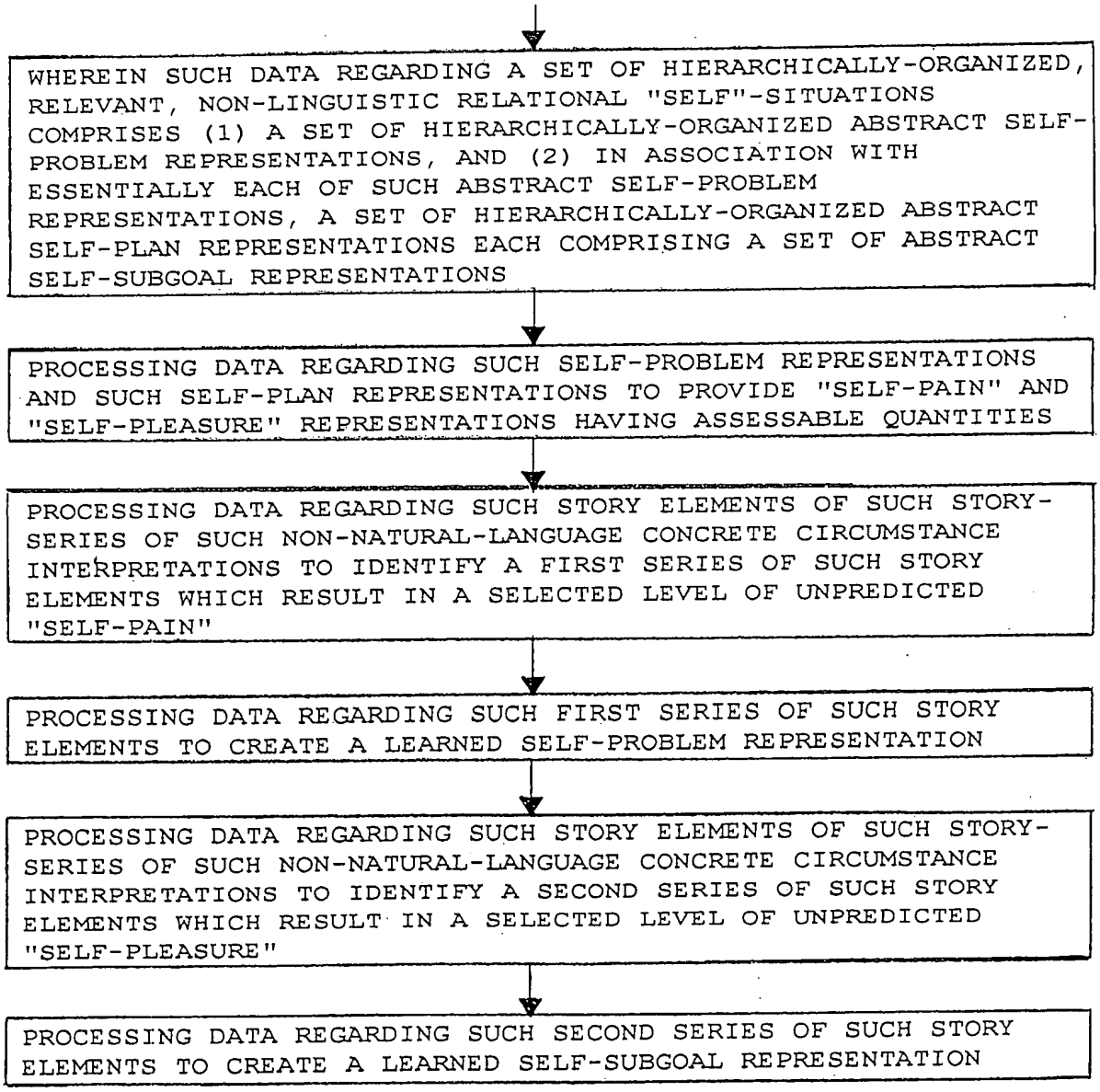

SUCH SET OF SELF-TENDENCY MARKERS HAVING A 1-TO-1 CORRESPONDENCE WITH SUCH SET OF SELF-TENDENCIES, ONE UNIQUE MARKER FROM SUCH SET OF SELF-TENDENCY MARKERS CORRESPONDING RESPECTIVELY WITH EACH SELF-TENDENCY OF SUCH SET OF SELF-TENDENCIES
SUCH SUBSET OF SELF-TENDENCIES BEING CONSTRUCTED AND ARRANGED TO PERMIT A DETERMINATION OF THE EXPECTED BEHAVIOR OF EACH SUCH OBJECT REPRESENTATION WITH RESPECT TO ANY MAPPABLE REPRESENTATION OF OTHER OBJECT REPRESENTATIONS FROM SUCH COLLECTION OF OBJECT REPRESENTATIONS
EACH SUCH SELF-TENDENCY CONSISTING ESSENTIALLY OF AN INSTRUCTION FOR SELF-BEHAVIOR (OF ANY FIRST OBJECT REPRESENTATION TO WHICH SUCH SELF-TENDENCY MAY BE ASSIGNED) ON THE CONDITION THAT ANY MAPPABLE REPRESENTATION OF OBJECT REPRESENTATIONS FROM SUCH COLLECTION, INCLUDING SUCH FIRST OBJECT REPRESENTATION, FROM THE VIEWPOINT OF SUCH FIRST OBJECT REPRESENTATION, IS INCLUDED IN A SPECIFIED SELF-RELATION SELECTED FROM A SET OF SELF-RELATIONS
WHERE EACH SELF-RELATION OF SUCH SET OF SELF-RELATIONS COMPRISES A SPECIFIED SPACE-TIME RELATION AMONG A SUCH FIRST OBJECT REPRESENTATION AND AT LEAST ONE SUCH OTHER OBJECT REPRESENTATION
EACH SUCH OTHER OBJECT REPRESENTATION BEING SPECIFIED AS A SUBSET OF A SET OF SELF-TENDENCY MARKERS, EACH SUCH SUBSET OF SELF-TENDENCY MARKERS CORRESPONDING TO THE SUBSET OF SELF-TENDENCIES ASSIGNED TO EACH SUCH OTHER OBJECT REPRESENTATION.

FIG. 9B-5


WHEREIN SUCH DATA REGARDING A SET OF HIERARCHICALLY-ORGANIZED, RELEVANT, NON-LINGUISTIC RELATIONAL "SELF"-SITUATIONS COMPRISES (1) A SET OF HIERARCHICALLY-ORGANIZED ABSTRACT SELF-PROBLEM REPRESENTATIONS, AND (2) IN ASSOCIATION WITH ESSENTIALLY EACH OF SUCH ABSTRACT SELF-PROBLEM REPRESENTATIONS, A SET OF HIERARCHICALLY-ORGANIZED ABSTRACT SELF-PLAN REPRESENTATIONS EACH COMPRISING A SET OF ABSTRACT SELF-SUBGOAL REPRESENTATIONS
PROCESSING DATA REGARDING SUCH SELF-PROBLEM REPRESENTATIONS AND SUCH SELF-PLAN REPRESENTATIONS TO PROVIDE "SELF-PAIN" AND "SELF-PLEASURE" REPRESENTATIONS HAVING ASSESSABLE QUANTITIES
PROCESSING DATA REGARDING SUCH STORY ELEMENTS OF SUCH STORY-SERIES OF SUCH NON-NATURAL-LANGUAGE CONCRETE CIRCUMSTANCE INTERPRETATIONS TO IDENTIFY A FIRST SERIES OF SUCH STORY ELEMENTS WHICH RESULT IN A SELECTED LEVEL OF UNPREDICTED "SELF-PAIN"
PROCESSING DATA REGARDING SUCH FIRST SERIES OF SUCH STORY ELEMENTS TO CREATE A LEARNED SELF-PROBLEM REPRESENTATION
PROCESSING DATA REGARDING SUCH STORY ELEMENTS OF SUCH STORY-SERIES OF SUCH NON-NATURAL-LANGUAGE CONCRETE CIRCUMSTANCE INTERPRETATIONS TO IDENTIFY A SECOND SERIES OF SUCH STORY ELEMENTS WHICH RESULT IN A SELECTED LEVEL OF UNPREDICTED "SELF-PLEASURE"
PROCESSING DATA REGARDING SUCH SECOND SERIES OF SUCH STORY ELEMENTS TO CREATE A LEARNED SELF-SUBGOAL REPRESENTATION

```
{                    } Unit Sagetalk; {                                    }

{ Purpose: ================================================ Owner: TFL ===
  This unit contains the code that generates story text, based on the
  arousal record in the story2 unit. All kinds of substitutions occur.
  History: ------------------------------------------------------------
  3/26/91 : Added Marty's sagetalk changes. RG
  7/24/91 : Mods made to reflect use of new synonym calls and to reflect
            spec changes per file SPTELNW3.1SP. - DS
  7/31/91 : Mods made to reflect new face emotional settings per Jay's
            spec, SAGEFACE.JAY. - DS
  Copyright (c) 1989-1991 Anthrobotics ======================================}
 {                    } Interface {                                         }
 Uses Crt,Story2,Types,Playback;
 FUNCTION  Describe_Story(Caps: Boolean): Integer;
 PROCEDURE Intro_Phrase(Which: Integer);
 PROCEDURE Then_Phrase(Which: Integer; Prefix,Suffix: Boolean);
 PROCEDURE Problem_Phrase(Which: Integer);
 PROCEDURE Strategy_Phrase(Which: Integer);
 PROCEDURE Decision_Phrase(Which: Integer);
 PROCEDURE NewDecision_Phrase(Which: Integer);
 PROCEDURE Plan_Phrase(Which: Integer);
 PROCEDURE NextNode_Phrase(Which: Integer);
 PROCEDURE Node_Phrase(Which: Integer);
 PROCEDURE Friend_Phrase(Which: Integer);
 PROCEDURE Last_Phrase(Which: Integer);
 PROCEDURE Describe_Arousals(Which:Integer);
 PROCEDURE Get Highest_Regs;
{                    } Implementation {                                    }
 Uses  General,NewTypes,Register,Face2,
       Terms4,Terms5,Terms6,
       DatList,Terms1,Terms2,Terms3,GoalStuff;

 Const UseCaps: Boolean = False;  {if true, then sound_and_Text upcases 1st char}
       Spoke  : Boolean = False;

 Type  Volume = (VSame,VLoud,VNorm,VSoft);
       Pitch  = (PSame,PHiHi,PHi,PNorm,PLo,PLoLo);
       Faces  = (FNorm,fSad,fGrim,fCTime);
 {-------------------------------------------------------------}
 PROCEDURE Output_Face_Command(Command: KeyWords; Parm1,Parm2,Parm3,Parm4:
ShortInt);
 Begin
      StoryFile.fCurrentCommand.Bytes[1]:=parm1;
      StoryFile.fCurrentCommand.Bytes[2]:=Parm2;
      StoryFile.fCurrentCommand.Bytes[3]:=Parm3;
      StoryFile.fCurrentCommand.Bytes[4]:=Parm4;
      OutPut('',Command);
 end; {output face change}
 {-------------------------------------------------------------}
 PROCEDURE Sound_And_Text(TheVolume: Volume; ThePitch: Pitch; S: Str150);
 Const VolumeLevels : Array[Volume] of Byte = (0,6,5,4);
       PitchLevels  : Array[Pitch]  of Byte = (0,8,7,6,5,4);
 Begin
      If UseCaps then S[1]:=UpCase(S[1]);
      Spoke:=True;
      UseCaps:=False;
      Save_Sound(0,VolumeLevels[TheVolume],PitchLevels[ThePitch],10);
      OutPut(S,Say);
 end; {sound and text}
 {-------------------------------------------------------------}
 PROCEDURE Show_Blink;
 Const Command : Array[1..6] of Keywords = (LEye,Reye,Display,Wait,Leye,Reye);
       Parm    : Array[1..6] of ShortInt = (100,100,0,125,-100,-100);
 Var   I       : integer;
 Begin
      For I:=1 to 6 do
          Output_Face_Command(Command[i],Parm[i],0,0,0);
 end; {blink}

 {-------------------------------------------------------------}
```

```
 PROCEDURE Drive_Face(Var Comms,Prms; Count : Byte);
 Var Commands : Array[1..1] of KeyWords absolute Comms;
     Parm     : Array[1..1,1..4] of ShortInt absolute Prms;
 Var I        : Integer;
 Begin
      For I:=1 to Count do
Output_Face_Command(Commands[i],parm[i,1],Parm[i,2],Parm[i,3],parm[i,4]);
 end; {Drive face}
 {-------------------------------------------------------------}
 PROCEDURE Show_Grim;
 Const Command : Array[1..5] of Keywords = (Eyes,Leye,Reye,Nose,Mouth);
       Parm    : Array[1..5,1..4] of ShortInt =
((-15,0,15,0),(0,-15,-10,0),(0,-15,-10,0),(10,0,0,0),(-10,-20,-10,0));
 Begin
     Drive_Face(Command,Parm,5);
 end; {grim}
 {-------------------------------------------------------------}
 PROCEDURE Show_Sad;
 Const Command : Array[1..4] of Keywords = (Eyes,Leye,Reye,Mouth);
       Parm    : Array[1..4,1..4] of ShortInt =
         ((0,20,0,7),(20,0,-25,0),(20,0,-25,0),(-5,-50,10,0));
 Begin
      Drive_Face(Command,Parm,4);
 end; {sad}
 {-------------------------------------------------------------}
 PROCEDURE Show_Fear;
 Const Command : Array[1..6] of Keywords = (Eyes,Leye,Reye,Mouth,Pupils,Nose);
       Parm    : Array[1..6,1..4] of ShortInt =
         ((25,20,25,0),(1,50,0,0),(1,50,0,0),(-25,-50,60,0),(3,0,0,0),(3,0,0,0));
 Begin
     Drive_Face(Command,Parm,6);
 end; {show fear}
 {-------------------------------------------------------------}
 PROCEDURE Show_Smile;
 Const Command : Array[1..4] of Keywords = (Mouth,Eyes,Leye,Reye);
       Parm    : Array[1..4,1..4] of ShortInt =
         ((5,35,20,0),(-10,0,5,0),(0,15,0,0),(0,15,0,0));
 Begin
     Drive_Face(Command,Parm,4);
 end; {Smile}
 {-------------------------------------------------------------}
 PROCEDURE Show_Wink;
 Const Command : Array[1..6] of Keywords = (REye,Leye,Display,Wait,Reye,LEye);
       Parm    : Array[1..6,1..4] of ShortInt =
((100,-20,0,0),(0,30,0,0),(0,0,0,0),(127,0,0,0),(-100,20,0,0),(0,-30,0,0));
 Begin
     Drive_Face(Command,Parm,6);
 end; {Wink}
 {-------------------------------------------------------------}
 PROCEDURE Move_Brows(Up: boolean);
 Const Scalar : Array[Boolean] of ShortInt = (-20,20);
 Begin
     Output_Face_Command(Leye,0,Scalar[up],0,0);
     Output_Face_Command(Reye,0,Scalar[up],0,0);
 end; {Move Brows}
 {-------------------------------------------------------------}
 PROCEDURE Tilt_Left_Brows(Up: boolean);
 Const Scalar : Array[Boolean,1..3] of ShortInt = ((-20,-20,10),(20,20,-10));
 Begin
     Output_Face_Command(Leye,0,Scalar[up,1],Scalar[Up,2],0);
     Output_Face_Command(Reye,0,Scalar[up,3],0,0);
 end; {tilt left brows}
 {-------------------------------------------------------------}
 PROCEDURE Look_Around;
 Const Command : Array[1..7] of Keywords =
(Eyes,Display,wait,eyes,display,wait,eyes);
       Parm    : Array[1..7,1..4] of ShortInt =
((0,0,0,-14),(0,0,0,0),(127,0,0,0),(0,0,0,28),(0,0,0,0),(127,0,0,0),(0,0,0,-14));
 Begin
     Drive_Face(Command,Parm,7);
 end; {look around}
 {-------------------------------------------------------------}
```

FIG. 9C-3

```
PROCEDURE Init_Face(What: Faces);
Const Default : Array[1..15] of ShortInt =
(15,15,-10,-10,0,4,1,1,10,0,10,-10,0,30,1);
Begin
     If What=fCTime then
          Move(Default,StoryFile.fInput.fData,SizeOf(Default))
     else Move(LastCompFace,StoryFile.fInput.fData,SizeOf(Default));
     OutPut('',AbsFace);
     If What=fGrim then Show_Grim else
        If What=fSad then Show_Sad;
end; {init face}
{-------------------------------------------------------------}
FUNCTION Describe_Story(Caps: Boolean): Integer;
Begin
     Spoke:=False;
     UseCaps:=Caps;
     With AStoryRec,StoryFile do
     Begin
          If (NowPorpoisePtr<>LastNowPorpoisePtr) then
             Begin
                  Problem_Phrase(AnyTerm);
                  If TheStratName<>'' then
                     Strategy_Phrase(Anyterm);
             end;
          If (HitSiNode<>LastStoryrec.HitSiNode) then
             If (PlanName<>'') and (SGName<>'') then
                 Plan_Phrase(Anyterm)
             else
          else If (SGName<>'') then Node_Phrase(Anyterm);
          If (NextSINode<>LastStoryrec.NextSiNode) and
             (NextSGName<>'') then
                  NextNode_Phrase(Anyterm);
          If (NowDSD<>LastStoryrec.NowDSD) then
                  NewDecision_Phrase(Anyterm);
          If Spoke then
                  Describe_Arousals(AnyTerm);
     end;
     Describe_Story:=StoryFile.fIOresult;
end; {describe story}
{-------------------------------------------------------------}
PROCEDURE Intro_Phrase;
Var  T1,T2,T3,T4: String[70];
Begin
     Init_Face(fCTime);
     UseCaps:=True;
     Which:=Determine_Which(Which,6);
     Case Which of
          1: Begin
                  T1:=Instantly_Terms(anyterm,false,false)+', ';
                  T2:='while'+In_Terms(AnyTerm,True,True)+MainFileName+', is ';
                  T3:='where it''s';
                  T4:=Really_Terms(True,True)+Happening_Terms(False,False)+'. ';
             end;
          2: Begin
                  T1:='Well, ';
                  T2:='I''m'+In_Terms(AnyTerm,True,True)+MainFileName+' and it''s
'+Today;
                  T3:=' as it''s all';
                  T4:=Really_Terms(True,True)+Happening_Terms(False,False)+'. ';
             end;
          3: Begin
                  T1:='It''s a dark and'+Really_Terms(true,true)+'stormy night,
'+UserName+', ';
                  T2:='and I''m'+In_Terms(AnyTerm,True,True)+MainFileName+', ';
                  T3:='as I'+Usually_Terms(True,True)+'am, ';
                  T4:='minding my own business.';
             end;
          4: Begin
                  T1:='You won''t believe this, ';
                  T2:='but so help me ';
                  T3:='every word is'+Really_Terms(True,True)+'true. ';
                  T4:='It''s '+Today+' and
```

```
 I''m'+In_Terms(AnyTerm,True,True)+MainFileName+'. ';
                end;
             5: Begin
                     T1:=Instantly_Terms(anyterm,false,false)+', ';
                     T2:='I''m'+In_Terms(AnyTerm,True,True)+MainFileName+' ';
                     T3:='on '+Today+' ';
                     T4:='and things are
'+becoming_Terms(false,false)+Really_Terms(True,True)+'bad. ';
                end;
             6: Begin
                     T1:=Instantly_Terms(anyterm,false,false)+', ';
                     T2:='it''s all'+Began_Terms(ANyTerm,True,True);
                     T3:='while I''m'+In_Terms(AnyTerm,True,True);
                     T4:=MainFileName+'. ';
                end;
        end;
        Sound_And_Text(VNorm,PHi,T1);
        Show_Blink;
        Sound_And_Text(VNorm,PNorm,T2);
        Show_Fear;
        Sound_And_Text(VNorm,PHi,T3);
        Sound_And_Text(VSoft,PNorm,T4);
        Show_Blink;
 end; {intro phrase}
 {-------------------------------------------------------}
 PROCEDURE Problem_Phrase;
 Var temp : String[80];
 Begin
        Which:=Determine_WHich(Which,7);
        Case Which of
             1: Begin
                     temp:='the'+problem_Terms(AnyTerm,True,True);
                     Sound_And_Text(VNorm,PHi,temp);
                     Look_Around;
                     Sound_And_Text(VSoft,PNorm,'that''s facing me
'+Instantly_Terms(Anyterm,False,True));
                     Show_Blink;
                     Sound_And_Text(VSoft,PHi,'is '+ProblemName+'. ');
                end;
             2: Begin
                     Sound_And_Text(VNorm,PHi,'I''m'+Really_Terms(True,True)+'trying
');
                     Look_Around;
                     Sound_And_Text(VSoft,PNorm,'to avoid '+ProblemName+'. ');
                     Show_Blink;
                end;
             3: Begin
                     Init_Face(FSad);
Sound_And_Text(VNorm,PNorm,'I''m'+Began_Terms(AnyTerm,True,True)+'to have a' +
                        Problem_Terms(AnyTerm,True,True));
                     Look_Around;
                     Sound_And_Text(VSoft,PNorm,'trying to avoid ' +ProblemName+'.
');
                end;
             4: Begin
                     If Caps then ProblemName[1] := UpCase(ProblemName[1]);
                     Sound_And_Text(VNorm,PHi,ProblemName+' is ');
                     Tilt_Left_Brows(True);
Sound_And_Text(VNorm,PNorm,Really_Terms(False,True)+'a'+Problem_Terms(AnyTerm,True
,True)+'for me. ');
                     Show_Blink;
                end;
             5: Begin
                     Init_Face(FSad);
                     temp:='talk about a' +Problem_Terms(AnyTerm,True,False)+'--';
                     Sound_And_Text(VNorm,PNorm,temp);
                     Show_Blink;
                     Output_Face_Command(Eyes,0,0,0,7);
                     Sound_And_Text(VNorm,PLo,'the one I am'
                        +Really_Terms(true,true)+'facing is '+ProblemName+'. ');
                end;
```

```
            6: Begin
                    Init_Face(FSad);
Sound_And_Text(VNorm,PHiHi,Instantly_Terms(AnyTerm,False,False)+', ');
                    Show_Blink;
                    Output_Face_Command(Eyes,0,0,0,7);
                    Sound_And_Text(VNorm,PLo,'I''m confronting a new
'+Problem_Terms(Anyterm,False,false)
                        +', '+ProblemName+'. ');
               end;
            7: Begin
                    Init_Face(FSad);
                    temp:=ProblemName+' is the'+problem_Terms(AnyTerm,True,True);
                    Sound_And_Text(VNorm,PNorm,temp);
                    Show_Blink;
                    Output_Face_Command(Eyes,0,0,0,7);
                    Sound_And_Text(VNorm,PLo,'that is'+
                        Becoming_Terms(true,true)+'an aggravation. ');
               end;
      end;
      UseCaps:=True;
 end; {problem phrase}
 {-------------------------------------------------------}
 PROCEDURE Strategy_Phrase;
 Begin
      Which:=Determine_Which(Which,6);
      If Which<>6 then Init_Face(fGrim);
      Case Which of
           1: Sound_And_Text(VNorm,PHi,'so, of course,
I'+Planned_To_Terms(AnyTerm,True,True)+TheStratName+'. ');
           2: Begin
                    Sound_And_Text(VNorm,PHi,'all I'+Really_Terms(True,True)+'care
about is ');
                    Output_Face_Command(Eyes,0,0,0,7);
                    Sound_And_Text(VNorm,PNorm,'that
I'+Planned_To_Terms(AnyTerm,True,True)+TheStratName+'. ');
              end;
           3: Begin
                    Sound_And_Text(VNorm,PHi,'to '+TheStratName+' ');
                    Sound_And_Text(VSoft,PNorm,'is what
I'+Really_Terms(True,True)+'am focused on. ');
              end;
           4: Begin
                    Sound_And_Text(VNorm,PHi,'I know if I can ' +TheStratName+' ');
                    Show_Blink;
                    Sound_And_Text(VSoft,PNorm,'I
will'+Usually_Terms(True,True)+'be all right'
                        +very_Soon_Terms(true,false)+'. ');
              end;
           5: Begin
                    Output_Face_Command(Eyes,0,0,0,-14);
                    Sound_And_Text(VNorm,PHi,'if I can only '+TheStratName+' ');
                    Output_Face_Command(eyes,0,0,0,14);
                    Sound_And_Text(VSoft,PNorm,'my'
+Problem_Terms(AnyTerm,True,True)+ 'will end. ');
              end;
           6: Begin
                    Output_Face_Command(Eyes,0,0,0,-14);
                    Sound_And_Text(VNorm,PHi,'but I
can'+Usually_Terms(True,True)+'handle it ');
                    Output_Face_Command(eyes,0,0,0,14);
                    Sound_And_Text(VSoft,PNorm,'if I can only '+TheStratName+'. ');
              end;
      end;
      UseCaps:=True;
 end; {strategyphrase}
 {-------------------------------------------------------}
 PROCEDURE Decision_Phrase;
 Begin
      Which:=Determine_Which(Which,7);
      If Which<5 then Init_Face(fNorm);
      Case Which of
```

```
          1: Begin
Sound_And_Text(VNorm,PHi,instantly_Terms(anyterm,False,False)+', ');
                    Sound_And_Text(VNorm,PNorm,'I am'+
Began_Terms(AnyTerm,True,True)+' to '+DfltDSDName+' ');
                    Show_Fear;
Sound_And_Text(VNorm,PLo,'while'+becoming_Terms(true,true)+'concerned');
                    Sound_And_Text(VSoft,PLo,'and'+Looking_For_Terms(True,True)+'a
plan. ');
               end;
          2: Begin
                    Sound_And_Text(VNorm,PNorm,'I''m getting my brain in gear ');
                    Show_Blink;
Sound_And_Text(VNorm,PLo,'and'+Began_Terms(AnyTerm,True,false)+' to
'+DfltDSDName+'. ');
               end;
          3: Begin
Sound_And_Text(VSoft,PNorm,'while'+Decision_Terms(AnyTerm,True,True));
                    Move_Brows(True);
                    Sound_And_Text(VSoft,PHi,'what else I might do, ');
                    Move_Brows(False);
                    Show_Blink;
                    Sound_And_Text(VNorm,PNorm,'I
am'+Began_Terms(AnyTerm,True,True)+' to '+DfltDSDName+'. ');
                    Show_Smile;
               end;
          4: Begin
                    Sound_And_Text(VNorm,PNorm,Right_Now_Terms(False,False)+', I
am');
                    Show_Blink;
                    Sound_And_Text(VNorm,PNorm,Began_Terms(AnyTerm,True,True)+' to
'+DfltDSDName+' ');
                    Show_Smile;
                    Output_Face_Command(Eyes,0,0,0,-14);
                    Sound_And_Text(VNorm,PLo,' and worrying less ');
                    Output_Face_Command(Eyes,0,0,0,14);
                    Sound_And_Text(VNorm,PLo,' while'
+Decision_Terms(AnyTerm,True,True)+'my ');
                    Move_Brows(True);
                    Sound_And_Text(VNorm,PLolo,'next step. ');
               end;
          5: Begin
                    Sound_And_Text(VNorm,PNorm,'so I''m');
                    Show_Blink;
                    Sound_And_Text(VNorm,PLo,Began_Terms(AnyTerm,True,True)+' to
'+DfltDSDName
                       +instantly_terms(anyterm,true,false)+'. ');
               end;
          6: Begin
Sound_And_Text(VNorm,PNorm,Instantly_Terms(anyterm,False,False)+
                       ', I''m '+Decision_Terms(AnyTerm,False,True));
                    Show_Blink;
                    Sound_And_Text(VNorm,PLo,'to start to '+DfltDSDName+' ');
                    Show_Smile;
               end;
          7: Begin
                    Sound_And_Text(VNorm,PNorm,'so, ');
                    Show_BLink;
Sound_And_Text(VNorm,PNorm,'while'+Looking_For_Terms(True,True)+'some plan, ');
                    Show_Smile;
                    Output_Face_Command(Eyes,0,0,0,-14);
Sound_And_Text(VNorm,PLo,'I''m'+Began_Terms(AnyTerm,True,True)+' to
'+DfltDSDName+'. ');
                    Move_Brows(True);
               end;
     end;
     UseCaps:=True;
end; {Decision phrase}
{-------------------------------------------------------}
PROCEDURE Plan_Phrase;
Begin
     Which:=Determine_Which(Which,8);
     Init_Face(FNorm);
```

```
      Case Which of
           1 : Begin
Sound_And_Text(VNorm,PHi,Instantly_Terms(anyterm,False,False)+', I am '+SGName+'
');
                    Tilt_Left_Brows(True);
                    Sound_And_Text(VNorm,PNorm,'and
I'+Planned_To_Terms(AnyTerm,True,True)+PlanName+'. ');
                    Show_Blink;
               end;
           2 : Begin
                    Sound_And_Text(VNorm,PHi,'I find myself '+SGName+' ');
                    Show_Blink;
                    Show_Grim;
Sound_And_Text(VNorm,PNorm,'and'+Planned_To_Terms(AnyTerm,True,True)+PlanName+'.
');
               end;
           3 : Begin
                    Sound_And_Text(VSoft,PLo,'so here I am, ');
                    Show_Blink;
                    Sound_And_Text(VSoft,PLo,SGName+' ');
                    Tilt_Left_Brows(True);
                    Sound_And_Text(VNorm,PNorm,'and
I'+Planned_To_Terms(AnyTerm,True,True)+PlanName+'. ');
               end;
           4 : Begin
                    Sound_And_Text(VSoft,PLo,'anyway, ');
                    Show_Wink;
                    Sound_And_Text(VSoft,PLo,'I am '+SGName+' ');
                    OutPut_Face_Command(Eyes,0,0,0,-7);
Sound_And_Text(VSoft,PNorm,'and'+Planned_To_Terms(AnyTerm,True,True)+PlanName+'.
');
               end;
           5 : Begin
                    Sound_And_Text(VNorm,PHi,'here I''m '+SGName+' ');
                    Tilt_Left_Brows(True);
                    Sound_And_Text(VNorm,PNorm,'but
I'+Planned_To_Terms(AnyTerm,True,True)+PlanName+' next. ');
                    Show_Blink;
               end;
           6 : Begin
Sound_And_Text(VSoft,PLo,Instantly_Terms(anyterm,False,False)+' I''m ');
                    Show_Blink;
                    Sound_And_Text(VSoft,PLo,SGName+' ');
                    Tilt_Left_Brows(True);
                    Sound_And_Text(VNorm,PNorm,'but next
I'+Planned_To_Terms(AnyTerm,True,True)+PlanName+'. ');
               end;
           7 : Begin
Sound_And_Text(VSoft,PLo,Instantly_Terms(anyterm,False,False)+', while I''m ');
                    Show_Wink;
                    Sound_And_Text(VSoft,PLo,SGName+', ');
                    OutPut_Face_Command(Eyes,0,0,0,-7);
Sound_And_Text(VSoft,PNorm,'I'+Planned_To_Terms(AnyTerm,True,True)+PlanName+'. ');
               end;
           8 : Begin
                    Sound_And_Text(VSoft,PNorm,'I
next'+Planned_To_Terms(AnyTerm,True,True)+PlanName+' ');
                    Show_Blink;
                    Sound_And_Text(VSoft,PLo,'from where I am now '+SGName+'. ');
                    OutPut_Face_Command(Eyes,0,0,0,-10);
               end;
      end;
      UseCaps:=True;
end; {plan phrase}
{-------------------------------------------------------}
PROCEDURE Node_Phrase;
Var St : String[40];
Begin
      Which:=Determine_Which(Which,7);
      Init_Face(fNorm);
      Case Which of
           1 : St:='I am';
```

```
            2 : St:='I find myself';
            3 : St:=Instantly_Terms(anyterm,false,false)+', I find myself';
            4 : St:='here I am';
            5 : St:=Instantly_Terms(anyterm,false,false)+', I am';
            6 : St:='I''m here';
            7 : St:='I''m';
      end; {case}
      If SGName[1]<>' ' then St:=St+' ';
      Sound_And_Text(VNorm,PHi,St+SGName+'. ');
      Move_Brows(True);
      UseCaps:=True;
 end; {node phrase}
  {-------------------------------------------------------}
 PROCEDURE NextNode_Phrase;
 Var St : STring[40];
 Begin
      Which:=Determine_Which(Which,6);
      Init_Face(fGrim);
      Case Which of
            1 : St := 'I';
            2 : St := 'but I';
            3 : St := 'so I';
            4 : St := 'I know I';
            5 : St := 'without a doubt, I';
            6 : St := 'obviously, I';
      end; {Case}
      Sound_And_Text(VLoud,PHi,St+Really_Terms(True,True));
      Sound_And_Text(VNorm,PHi,Need_To_Be_Terms(AnyTerm,False,True)+NextSGName+'.
');
      UseCaps:=True;
 end; {next node phrase}
 {-------------------------------------------------------}
 PROCEDURE NewDecision_Phrase;
 Begin
      Which:=Determine_Which(Which,8);
      Init_Face(FNorm);
      Case Which of
            1 : Begin
                  OutPut_Face_Command(Eyes,0,0,0,8);
                  Sound_And_Text(VNorm,PHi,'I am'+Began_Terms(Which,True,True));
                  Sound_And_Text(Vnorm,PNorm,'to '+NowDSDName);
                  Show_Blink;
                  Show_Smile;
                  Sound_And_Text(VNorm,PNorm,'
while'+hoping_Terms(true,true)+'for the best. ');
                end;
            2 : Begin
                  Sound_And_Text(VLoud,PHi,'so, ');
                  Show_Blink;
Sound_And_Text(VNorm,PNorm,'I''m'+Began_Terms(Which,True,True));
                  Sound_And_Text(Vnorm,PHi,'to '+NowDSDName+'. ');
                end;
            3 : Begin
                  Sound_And_Text(VSoft,PLo,'to get to where I''m going, ');
                  Look_Around;
Sound_And_Text(VNorm,PNorm,'I''m'+Began_Terms(Which,True,True));
                  Tilt_Left_Brows(True);
                  Sound_And_Text(Vnorm,PHi,'to '+NowDSDName+'. ');
                end;
            4 : Begin
                  Sound_And_Text(VSoft,PNorm,'to move along my plan, ');
                  OutPut_Face_Command(Eyes,0,0,0,8);
                  OutPut_Face_Command(Eyes,0,0,0,-12);
Sound_And_Text(VNorm,PNorm,'I''m'+Began_Terms(Which,True,True));
                  Sound_And_Text(Vnorm,PHi,'to '+NowDSDName+'. ');
                end;
            5 : Begin
                  Tilt_Left_Brows(True);
Sound_And_Text(VNorm,PNorm,'I''m'+Began_Terms(Which,True,True));
                  Sound_And_Text(VNorm,PHi,'to '+NowDSDName+', ');
                  Show_Smile;
```

```
 Sound_And_Text(VNorm,PNorm,Hoping_Terms(False,True)+'to get to where I''m going.
');
                end;
            6 : Begin
                    Show_Blink;
Sound_And_Text(VNorm,PNorm,'I''m'+Began_Terms(Which,True,True));
                    Sound_And_Text(VNorm,PLo,'to '+NowDSDName+', ');
                    Look_Around;
                    Sound_And_Text(VNorm,PNorm,Hoping_Terms(False,True)+'my plan
works out. ');
                end;
            7 : Begin
                    Sound_And_Text(VLoud,PHi,'anyway,
'+Hoping_Terms(False,True)+'for the best, ');
                    Show_Blink;
Sound_And_Text(VNorm,PNorm,'I''m'+Began_Terms(Which,True,True));
                    Sound_And_Text(VNorm,PLo,'to '+NowDSDName+'. ');
                end;
            8 : Begin
                    Sound_And_Text(VLoud,PHi,'looking ahead, ');
                    Show_Blink;
Sound_And_Text(VNorm,PNorm,'I''m'+Began_Terms(Which,True,True));
                    Sound_And_Text(VNorm,PLo,'to '+NowDSDName+'. ');
                end;
      end;
      UseCaps:=True;
 end; {new decision}
 {-------------------------------------------------------}
 PROCEDURE Then_Phrase;
 Var St : String[20];
 Begin
      UseCaps:=True;
      Case Determine_Which(Which,7) of
           1 : St:='Then';
           2 : St:='Later';
           3 : St:='After a while';
           4 : St:='After that';
           5 : St:='Next';
           6 : St:='Then after that';
           7 : St:='Next thing';
      end;
      OutPut(Spacing[Prefix]+st+','+Spacing[Suffix],Say);
      UseCaps:=False;
 end; {then phrase}
 {-------------------------------------------------------}
 PROCEDURE Friend_Phrase;
 Var T1,T2 : String[80];
 Begin
      UseCaps:=True;
      Which:=Determine_Which(AnyTerm,6);
      If Which<4 then
      Begin
           Case Which of
                1: Begin
                        Init_Face(fCTime);
                        T1:='your'+Friend_Terms(AnyTerm,True,True)+HeroName+' ';
                        T2:='could'+Really_Terms(True,True)+'use your help and
advice, '+UserName+'. ';
                   end;
                2: Begin
                        Init_Face(fNorm);
                        T1:='I tell you, '+UserName+', ';
                        T2:='your'+Friend_Terms(AnyTerm,True,True)+
                           HeroName+' is'+Really_Terms(True,True)+'in trouble. ';
                   end;
                3: Begin
                        Init_Face(fNorm);
                        T1:='listen '+UserName+', ';
                        T2:='this time your'+Friend_Terms(AnyTerm,True,True)+
                            HeroName+' is'+Really_Terms(True,True)+'in a pickle.
';
                   end;
```

```
            end; {Case}
            Show_Wink;
            Sound_And_Text(VNorm,PLo,T1);
            Show_Blink;
            Show_Smile;
            Sound_And_Text(VNorm,PNorm,T2);
      end;
      UseCaps:=True;
 end; {Friend phrase}
 {-------------------------------------------------------}
 PROCEDURE Last_Phrase;
 Const Words: Array[1..5] of String[12] =
       ('at last,','finally,','so finally,','in the end,','so at last,');
       EndPhrase : Array[0..1] of String[20] =
       (' I reached my goal. ',' my problem ended. ');
 Begin
      Which:=Determine_Which(Which,5);
      UseCaps:=True;
      Init_Face(fCTime);
      Case PalDead of
           False : Begin
                      If SGName<>'' then
                         Sound_And_Text(VSoft,PNorm,Words[Which]+' I''m
'+SGName+'. ')
                      else
Sound_And_Text(VSoft,PNorm,Words[WHich]+EndPhrase[Random(2)]);
                      Describe_Arousals(AnyTerm);
                   end;
           True  : Sound_And_Text(VNorm,PNorm,Words[Which]+' I''m dead. ');
      end;
      UseCaps:=True;
 end; {last phrase}
 {-------------------------------------------------------}
 PROCEDURE Get_Highest_Regs;
 Var Element,
     Where,
     Count     : integer;
     Regs      : ByteArray;
     St        : String[80];
 Begin
      FillChar(Top3Arousals,SizeOf(Top3Arousals),0);
      Move(AStoryRec.Arousals,Regs,SizeOf(Regs));
      Regs[Ord(PainJoyRate)] :=0;  {NEVER talk about Pain, and always talk joy
(below)}
      FillChar(Top3Regs,SizeOf(Top3Regs),0);
      TopRegVal:=0;  {needed in story}
      AStoryRec.ValidRegCount:=0;
      (* First, get top 3 registers  --*)
      With AStoryRec do
      For Count:=1 to 3 do
      Begin
           Where:=Find_Largest_Buffer_Value(Regs[9],5);
           If (Where<>0) and (Regs[8+Where]<>0) then
           Begin
                Inc(ValidRegCount);
                Where:=Where+8;
                Top3Regs[ValidRegCount]:=Where;
                If Regs[Where]>TopRegVal then
                   TopRegVal:=Regs[where];
                Regs[Where]:=0;
           end
      end;
      If (AStoryRec.ValidRegCount<3) and (Regs[Ord(TotalArousal)]>0) then
      Begin
           Inc(AstoryRec.ValidRegCount);
           Top3Regs[AStoryRec.ValidRegCount]:=Ord(TotalArousal);
      end;

      (*-- Now convert top 3 regs into strings... --*)
      With AStoryRec do
      For Count:=1 to ValidRegCount do
      Begin
```

```
            St:=Amount_Of(Top3Regs[Count],0,255);
            Case Top3Regs[Count] of
                 8 : Top3Arousals[Count]:=St+Fear_Terms(AnyTerm,False,False);
                 9 : Top3Arousals[Count]:=St+Surprise_Terms(AnyTerm,False,False);
                10 :
Top3Arousals[Count]:=St+Disappointed_Terms(AnyTerm,False,False);
                11 : Top3Arousals[Count]:=St+Frustrated_Terms(AnyTerm,False,False);
                12 : Top3Arousals[Count]:=St+Hopeless_Terms(AnyTerm,False,False);
                14 : Top3Arousals[Count]:=St+Anxious_Terms(AnyTerm,False,False);
            end;
       end;
 end; {Get highest regs}
 {-------------------------------------------------------}
 PROCEDURE Describe_Arousals;
 Var Next     : Integer;
     Num      : Integer;
     TempStr : String[150];
 Begin
      Which:=Determine_Which(Which,3);
      If AStoryRec.ValidRegCount>0 then
      Begin
           Init_Face(fCTime);
           Case Which of
                1 : TempStr:='I'+Felt_Terms(AnyTerm,True,True)+Top3Arousals[1];
                2 : TempStr:='Mainly,
I'+Felt_Terms(AnyTerm,True,True)+Top3Arousals[1];
                3 : Begin
                         OutPut_Face_Command(Eyes,0,0,0,7);
                         TempStr:='I''ve noticed I'+Felt_Terms(AnyTerm,True,True)+
Top3Arousals[1];
                    end;
           end;
           Sound_And_Text(VSoft,PHi,TempStr+'. ');
      end;
      If (AStoryRec.Arousals[13]<127) then
         Sound_And_Text(VNorm,PNorm,'I''m also feeling '+
           Amount_Of(AStoryRec.Arousals[13],0,127)+
           Better_Terms(anyterm,false,false)+'. ');
 end; {Describe arousals}
 {-------------------------------------------------------}
 end.


 {                  } Unit Story2; {                                    }
{ ═══ Purpose: ═══════════════════════════════════════════════════════ }
  This unit controls the output of story text in record mode, and handles
  the real time changes to the face if telling a story in realtime mode.
 ═══ History: ═══════════════════════════════════════════════════════
  3/26/91 : Last Ctime of story, only output conclusion phrase. RG
  3/27/91 : If RiNum=AnyRi in convert_ri routine, kick out 1st SE.Ri. RG
  6/26/91 : Added arousal output to story file. This is to let the PAL
            talk 2 way about arousals in story files.
 ═══ Copyright (c) 1989-1991 Anthrobotics ═══════════════════════════ }
{                  } Interface {                                       }
 Uses  Types,BlockFile,PlayBack,SiStuff,DatList,newTypes,General;
 Type  StoryRecord = Record
                     HitSiNode,
                     NextSINode,
                     NowStratPtr,                {Set in goalstuf: What now
porpoise}
                     NowDsd,                     {Set in actions: Deep Thought}
                     DfltDsd       : PointPtr;   {Set in actions: Deep Thought}
                     ArousalLevel  : integer;    {Set in SxStuff:
Incr_Arousal_Regs}
                     HitPlanRow,                 {Set in }
                     HitPlanNum    : Byte;       {Set in GoalStuf
Find_Strategy/Advanced_On_Plan}
                     Arousals      : ByteArray;  {Set in face2: Make_Comp_Face}
                     ValidRegCount : Byte;       {Set in sagetalk}
       end; {storyrecord}
 Const MinimumPain  : Integer = 20;              {Set in options unit}
       FearQuitLevel: Integer = 90;              {Set in options unit}
```

FIG. 9C-12


```
        MinimumJoy    : Integer = 90;
        Top3Regs      : Array[1..3] of byte= (0,0,0);
        TopRegVal     : Byte = 0;
        Top3Arousals  : Array[1..3] of String[30] = ('','','');
 Const  ProblemName   : String[80] = '';   {set in goalstuf: What_Now_Porpoise}
        TheStratName  : String[80] = '';   {set in goalstuf: What_Now_Porpoise}
        PlanName      : String[80] = '';   {set in goalstuf: Find_Strategy}
        SGName        : String[80] = '';   {determined real time}
        DfltDsdName   : String[80] = '';   {determined real time}
        NowDsdName    : String[80] = '';   {determined real time}
        NextSGName    : String[80] = '';   {determined real time}
 Var    AStoryRec,
        LastStorYRec : StoryRecord;
 PROCEDURE Try_To_Save_Story;
 FUNCTION  Begin_Story(Var BeginStory: Boolean): Integer;
 FUNCTION  Conclude_Story: Boolean;
 PROCEDURE Terminate_Run;
 PROCEDURE Convert_Terms;
 PROCEDURE Save_Sound(Tone,Volume,Pitch,Speed: Integer);
 FUNCTION  OutPut(S: Str150; Command: KeyWords): Integer;
 PROCEDURE Expand_Synonyms(Var S: String; MaxLen: Word);
 PROCEDURE Convert_Abstract_Ris_To_Pfs(Var Source: string);
 PROCEDURE Convert_Ri_To_Rx(Var Source: String);
{                    } Implementation {                                    }
 Uses Terms6,Terms5,Terms4,Terms2,Terms3,Terms1,
      SageTalk,Face2,RiStuff,Strucs,
      Printer,GrTool,GoalStuff,DsdStuff,Focus2,SxStuff,
      Relation,RxStuff1;
 {-------------------------------------------------------------}
 PROCEDURE Save_Sound(Tone,Volume,Pitch,Speed: Integer);
 Begin
      StoryFile.fCurrentCommand.Bytes[1]:=Tone;
      StoryFile.fCurrentCommand.Bytes[2]:=Volume;
      StoryFile.fCurrentCommand.Bytes[3]:=Pitch;
      StoryFile.fCurrentCommand.Bytes[4]:=Speed;
      OutPut('',Sounds);
 end; {save sound}
 {-------------------------------------------------------------}
 FUNCTION OutPut(S: Str150; Command: KeyWords): INteger;
 Var I : Integer;
     Dont : Boolean;
 Begin
      Dont:=False;
      If S<>'' then
         StoryFile.fInput.fStr:=S;
      StoryFile.fCurrentCommand.Command:=Command;
      If (StoryFile.fFileOpen) then
           OutPut:=StoryFile.mPut_Command
      else OutPut:=0;
      If OkToTalk then PlayBack_Command(storyfile,i,Dont);
 end; {Output}
 {-------------------------------------------------------------}
 PROCEDURE Save_Composite_Face;
 Var Speaking : Boolean;
 Begin
      Speaking:=OkToTalk;
      OkTotalk:=True;
      OutPut('',Display); {Force first face update; compface was built in Face2}
      OkTotalk:=Speaking;
 end; {save composite face}
 {-------------------------------------------------------------}
 FUNCTION Enough_Fear_For_A_Good_Story: Boolean;
   {This routine tests to see if the genetic fear level of the current
    goal is greater than some settable minimum. If not the brain isnt
    scared enough to tell a good story, and it should stop talking.}
 Begin
      Enough_Fear_For_A_Good_Story:=
         SubGoalPtr(NowPorpoisePtr)^.ASubGoal.Arousal>FearQuitLevel;
 end; {enough fear for a good story}

 {-------------------------------------------------------------}
```

```
PROCEDURE Test_Arousal_MinMax;
  {This routine tests to see if the current arousal level is less than
   the lowest level of arousal so far, and if it exceeds the highest level
   of arousal so far. In either case, it saves the min and max.}
Begin
     If AStoryrec.ArousalLevel>ArousalMax then
        ArousalMax:=AStoryRec.ArousalLevel;
     If AStoryrec.ArousalLevel<ArousalMin then
        ArousalMin:=AStoryRec.ArousalLevel;
end; {Test arousal minmax}
{-------------------------------------------------------------}
PROCEDURE Save_Arousal_Info;
  {This routine stores the 3 highest arousal registers into the
   current storyfile if the storyfile is open.}
Begin
     StoryFile.fCurrentCommand.Command:=ArousalInfo;
     Move(Top3Regs,StoryFile.fCurrentCommand.Bytes,3);
     StoryFile.fCurrentCommand.Bytes[4]:=TopRegval;
     StoryFile.mPut_Command;
end; {save arousal info}
{-------------------------------------------------------------}
PROCEDURE Try_To_Save_Story;
Var FaceShown  : Boolean;
    BeginStory,
    Caps,
    ScaredEnuf : Boolean;
    I          : integer;
Begin
     BeginStory:=False;
     FaceShown:=True;
     If Not OKToTalk then
          Save_Composite_Face
     else FaceShown:=False;
     Convert_Terms;
     With AStoryRec do
     If ((StoryFile.fValidStory) or
        (Begin_Story(BeginStory)=0)) then
     Begin
          FaceShown:=BeginStory;
          StoryFile.fIOresult:=0;   {Clear former errors}
          If (StoryFile.fFileOpen) then
          Begin
               StoryFile.fCurrentCommand.Command:=CTimeMarker;
               StoryFile.fCurrentCommand.Long:=NowTime;
               StoryFile.mPut_Command;
          end;
          Inc(StoryFile.fCTimesSaved);
          ScaredEnuf:=Enough_Fear_For_A_Good_Story;
          If (NowPorpoisePtr<>LastNowPorpoisePtr) or
             (AStoryRec.HitSiNode<>LastStoryRec.HitSINode) then
             Begin
                  If ScaredEnuf then
                     Begin
                          FaceShown:=Not BeginStory;
                          If Not BeginStory then
                             Then_Phrase(AnyTerm,True,True);
                          If (Not PalDead) then
                             If Describe_Story(BeginStory)=0 then;
                     end;
                  If BeginStory then
                     Friend_Phrase(AnyTerm);
             end;
          Test_Arousal_MinMax;
          If StoryFile.fFileOpen then
             Save_Arousal_Info;
          If (Not ScaredEnuf) or (PalDead) then
             FaceShown:=Conclude_Story;
          LastStoryRec:=AStoryRec;
     end;
     If Not FaceShown then
        Save_Composite_Face;
end; {Try to save story}
```

```
{-------------------------------------------------------------}
FUNCTION Begin_Story(Var BeginStory: Boolean): Integer;
Var Which,
    Result   : integer;
Begin
     With StoryFile do
     If (AStoryRec.ArousalLevel>=MinimumPain) then
          Begin
               BeginStory:=True;
               fValidStory:=True;
               FillChar(LastStoryRec,SizeOf(LastStoryRec),255);
               ArousalMax:=AStoryRec.ArousalLevel;
               ArousalMin:=AStoryRec.ArousalLevel;
               If (OutPut('',TopOfStory)=0) then
                  Begin
                       Intro_Phrase(Which);
                       StoryFile.fIOResult:=0;
                  end
               else Terminate_Run;
               Begin_Story:=0;
               FillChar(LastStoryRec,SizeOf(LastStoryRec),0);
          end
     else Begin_Story:=-1;
end; {begin story}
{-------------------------------------------------------------}
FUNCTION Conclude_Story: Boolean;
Begin
     Last_Phrase(AnyTerm);
     OutPut('',EndOfStory);
     StoryFile.fValidStory:=False;
     Conclude_Story:=True;
end; {conclude story}
{-------------------------------------------------------------}
PROCEDURE Terminate_Run;
Begin
     If StoryFile.mClose=0 then;
     {Stop the run...}
     {Show error}
end; {terminate run}
{-------------------------------------------------------------------------}
FUNCTION Get_Ri_To_Talk_About(Var Source: Str150; Var Where: Integer): String;
Var Next,
    Temp : Str150;
    EndPos : Integer;
Begin
     Where:=Pos('[',Source);
     If Where<>0 then
        Begin
             EndPos:=Pos(']',Source);
             If EndPos=0 then EndPos:=Succ(Length(Source));
             Get_Ri_To_Talk_About:=Copy(Source,Where+1,EndPos-Where-1);
             Delete(Source,Where,EndPos-Where+1);
        end
     else Get_Ri_to_Talk_About:='';
end; {Get ri to talk about}
{-------------------------------------------------------------------------}
FUNCTION Convert_Ri_To_Pf(Var Source: String): Boolean;
Var AnRi : RiPtr;
    Num,fNum: Integer;
Begin
     AnRi:=RiPtr(RiList.Whose_Name(Source,true,num,fnum));
     If AnRi<>Nil then
          AnRi^.Get_Best_PF_Match(Source)
     else Source:='';
     Convert_Ri_To_Pf:=Source<>'';
end; {convert ri to pf}
{-------------------------------------------------------------------------}
PROCEDURE Convert_Abstract_Ris_To_Pfs(Var Source: string);
Var I,Where : Integer;
    RiOrPF  : String[20];
Begin
     Where:=0;
```

```
      RiOrPf:=Get_Ri_To_Talk_About(Source,Where);
      While (RiOrPf<>'') do
      Begin
           If Convert_Ri_To_Pf(RiOrPf) then
              Insert(RiOrPf,Source,Where);
           RiOrPf:=Get_RI_To_Talk_About(Source,Where);
      end;
end; {convert abstract ris to pfs}
{-------------------------------------------------------------------------}
PROCEDURE Lower_Case(Var Source: Str150);
Var I : integer;
Begin
      For I:=1 to Length(Source) do
          If (Source[i]>'@') and
             (Source[i]<'[') then Inc(Source[i],32);
end; {lower case}
{-------------------------------------------------------------------------}
FUNCTION Determine_Table(Var S: String): Integer;
Var Temp,Err: integer;
    SubSt   : String[10];
Begin
      Temp:=0;
      While (S[temp+1] in ['0'..'9']) and (temp<4) do Inc(Temp);
      SubSt:=Copy(S,1,Temp);
      Delete(S,1,Temp);
      If SubSt<>'' then
         Begin
              Val(SubSt,Temp,Err);
              If Err<>0 then Temp:=0;
         end
      else Temp:=0;
      Determine_Table:=Temp;
end; {determine table}
{-------------------------------------------------------------------------}
FUNCTION Valid_Token(Var S: String; Var Table: Integer): Boolean;
Var TempSt: String;
Begin
      Table:=-1;
      If S[2] in ['0'..'9'] then
         Begin
              Delete(S,1,1);
              Table:=Determine_Table(S);
         end;
      Valid_Token:=Table>-1;
end; {valid token}
{-------------------------------------------------------------------------}
PROCEDURE Get_Syn1(Var S: STring; Table: Integer);
Var St: STring[100];
    Suffix : Boolean;
Begin
      Suffix:=False;
      Case Table of  {tables 0..10}
           0 : St:=Tired_Terms(false,suffix);
           1 : St:=Signif_Terms(false,suffix);
           2 : St:=A_Bit_Terms(false,suffix);
           3 : St:=Becoming_Terms(false,suffix);
           4 : St:=Possibly_Terms(false,suffix);
           5 : St:=Very_Much_Terms(AnyTerm,false,suffix);
           6 : St:=At_The_Terms(false,suffix);
           7 : St:=Right_By_The_Terms(false,suffix);
           8 : St:=Near_The_Terms(false,suffix);
           9 : St:=Far_From_The_Terms(false,suffix);
          10 : St:=Very_Soon_Terms(false,suffix);
          11 : St:=Moving_Terms(false,suffix);
      end;
      S:=St+S;
end; {Get syn 1}
{-------------------------------------------------------------------------}
PROCEDURE Get_Syn2(Var S: STring; Table: Integer);
Var ST : String[100];
    Suffix : Boolean;
Begin
```

```
     Suffix:=False;
     Case Table of
          12 : St:=Toward_The_Terms(false,suffix);
          13 : St:=From_The_Terms(false,suffix);
          14 : St:=Boredom_Terms(false,suffix);
          15 : St:=Better_Terms(AnyTerm,false,suffix);
          16 : St:=Little_Terms(AnyTerm,False,suffix);
          17 : St:=Sort_Of_Terms(AnyTerm,False,suffix);
          18 : St:=Usually_Terms(False,suffix);
          19 : St:=Move_Terms(false,suffix);
          20 : St:=Picnic_Terms(False,suffix);
          21 : St:=Pretty_Much_Terms(AnyTerm,False,suffix);
          22 : St:=A_Lot_Terms(AnyTerm,False,suffix);
     end;
     S:=St+S;
end; {Get syn 2}
{-------------------------------------------------------------------------}
PROCEDURE Get_Syn3(Var S: STring; Table: Integer);
Var suffix  : Boolean;
    St      : String[100];
Begin
     Suffix:=False;
    { suffix:=(S='') or (s[1]<>' '); }
     Case Table of  {Tables 21..30}
          23 : St:=Looking_For_Terms(False,suffix);
          24 : St:=Really_Terms(False,suffix);
          25 : St:=Find_Terms(False,suffix);
          26 : St:=Arousal_Terms(False,suffix);
          27 : St:=Fleeing_Terms(False,suffix);
          28 : St:=Happening_terms(False,suffix);
          29 : St:=Staying_Still_Terms(False,suffix);
          30 : St:=Escape_Terms(false,suffix);
          31 : St:=Eat_Terms(False,suffix);
          32 : St:=Rest_Terms(False,suffix);
          33 : St:=Nearby_Terms(False,suffix);
          34 : St:=Play_Terms(False,suffix);
     end;
     S:=St+S;
end; {Get syn 3}
{-------------------------------------------------------------------------}
PROCEDURE Insert_Synonym(Var S: STring; Table: Integer);
Begin
     If Table>22 then
          Get_Syn3(S,Table)
     else If Table>11 then
          Get_Syn2(S,Table)
     else Get_Syn1(S,Table);
end; {insert synonym}
{-------------------------------------------------------------------------}
PROCEDURE Expand_Synonyms(Var S: String; Maxlen: Word);
Var TokenPos, Table: Integer;
    TempSt         : String;
Begin
     Convert_Abstract_Ris_To_Pfs(S);
     TempSt:='';
     Repeat
          TokenPos:=Pos('@',S);
          If (TokenPos>0) then
             Begin
                  TempSt:=TempSt+Copy(S,1,TokenPos-1);
                  Delete(S,1,TokenPos-1);
                  If Valid_Token(S,Table) then
                       Insert_Synonym(S,Table)
                  else
                       Begin
                            TempSt:=TempSt+'@';
                            Delete(S,1,1);
                       end;
             end
     Until TokenPos=0;
     TempSt:=TempSt+S;
     S:=Copy(TempSt,1,MaxLen);
```

```
end; {expand synonyms}
{-------------------------------------------------------------------------}
PROCEDURE Convert_Ri_To_Rx(Var Source: String);
{ Purpose:
  Converts Ris in user identifier names to the actual Rxs that were used
  in SiMod and decision making.
  History:
  03/27/91 : If RiNum=AnyRi, kick out first SE.Ri.
}
Var RxName,
    RiName          : Str80;
    RiNum,
    RxCount,
    Where,I         : Integer;
    Pt              : PointPtr;
Begin
      RxName:='';
      RiNum:=0;
      RiName:=Get_Ri_To_Talk_About(Source,Where);  {Strip ri from identfier}
      While RIName<>'' do
      Begin
           Pt:= RiList.Whose_Name(RiName,True,RiNum,i);
           If Pt<>Nil then
               Begin
                    RxCount:=RiPtr(Pt)^.Get_Best_Rx_Match(Sxx^,RxName);
                    If RxCount=1 then
                       Insert(RxName,Source,WHere)
                    else If (Convert_Ri_To_Pf(RIName)) then
                            Begin
                                 Insert(RiName,Source,Where);
                                 RxName:=RIName
                            end
                    else Insert('<UNKNOWN>',Source,Where);
               end
           else
                If (UpCase_String(RiName)='HIT') then
                Begin
RxName:=BrainRow^.SxRowRec.RelRow^.SxRowRec.Kind^.RxRec.Name;
                    Insert(RxName,Source,Where);
                end
           else
               Insert('<NOHIT>',Source,Where);
           RiName:=Get_Ri_To_Talk_About(Source,Where);  {Strip ri from identfier}
      end;
      Expand_Synonyms(Source,255);
end; {Convert ri to rx}
{-------------------------------------------------------------}
PROCEDURE Convert_Terms;
Var S : String[150];
    P : Pointer;
Begin
{DEBUG
      Put_Text('Time: '+Int_Str(NowTime,1),10,10,15,True);
DEBUG}
     Convert_Ri_To_Rx(ProblemName);
     If (AStoryRec.HitSINode<>Nil) then
          Begin
               SGName:=AStoryRec.HitSiNode^.Who_Am_I^;
               If SGName<>'' then
                  Convert_Ri_to_Rx(SGName)
          end
     else SGName:='';
     If AStoryRec.NextSINode<>Nil then
          Begin
               NextSGName:=AStoryRec.NextSINode^.Who_Am_I^;
               If NextSGName<>'' then
                  Convert_Ri_To_Rx(NextSGName)
          end
     else NextSGName:='';
     If (AStoryRec.NowStratPtr<>Nil) then  {Plan/Strategy names already set in
actions}
          Begin
```

```
                If TheStratName<>'' then
                   Convert_Ri_To_Rx(TheStratName);
                If PlanName<>'' then
                   Convert_Ri_To_Rx(PlanName);
           end;

      If AStoryRec.DfltDsd<>Nil then
           Begin
                DfltDsdName:=AStoryRec.DfltDsd^.Who_Am_I^;
                If DfltDsdName<>'' then
                   Convert_Ri_To_Rx(DfltDsdName);
           end
      else DfltDsdName:='';

      If AStoryRec.NowDsd<>Nil then
           Begin
                NowDsdName:=AStoryRec.NowDsd^.Who_Am_I^;
                If NowDsdName<>'' then
                   Convert_Ri_To_Rx(NowDsdName);
           end
      else NowDsdName:='';
      Get_Highest_Regs;

 {DEBUG
      With AStoryRec do
      Begin
           Writeln(lst,'------[',NowTime,']----------------------------------');
           Writeln(Lst,'Problem : ',problem,' Ptr =
',seg(nowporpoiseptr^),':',ofs(nowporpoisePtr^));
           Writeln(Lst,'Strat   : ',Thestrat,' Ptr =
',seg(Strat^),':',ofs(Strat^));
           Writeln(Lst,'Plan    : ',Plan);
           Writeln(Lst,'SINode  : ',SG,' Ptr = ',seg(SiNode^),':',ofs(SiNode^));
           Writeln(Lst,'NextNode: ',NextSG,' Ptr =
',seg(NextNode^),':',ofs(NextNode^));
           Writeln(Lst,'DfltDSD : ',Story2.DfltDSD,' Ptr =
',seg(AStoryRec.DfltDSD^),':',ofs(AstoryRec.DfltDsd^));
           Writeln(Lst,'NowDSD  : ',Story2.NowDSD,' Ptr =
',seg(AStoryRec.NowDSD^),':',ofs(AstoryRec.NowDsd^));
           Writeln(Lst);
      end;
  DEBUG}

 end; {convert terms}

 {-------------------------------------------------------------------------}

{                    Unit Initialization                                   }

 end. {story2 unit}
```

FIG. 10A

REPRESENTATIVE FLOW CHART FOR COGNITIVE LEARNING

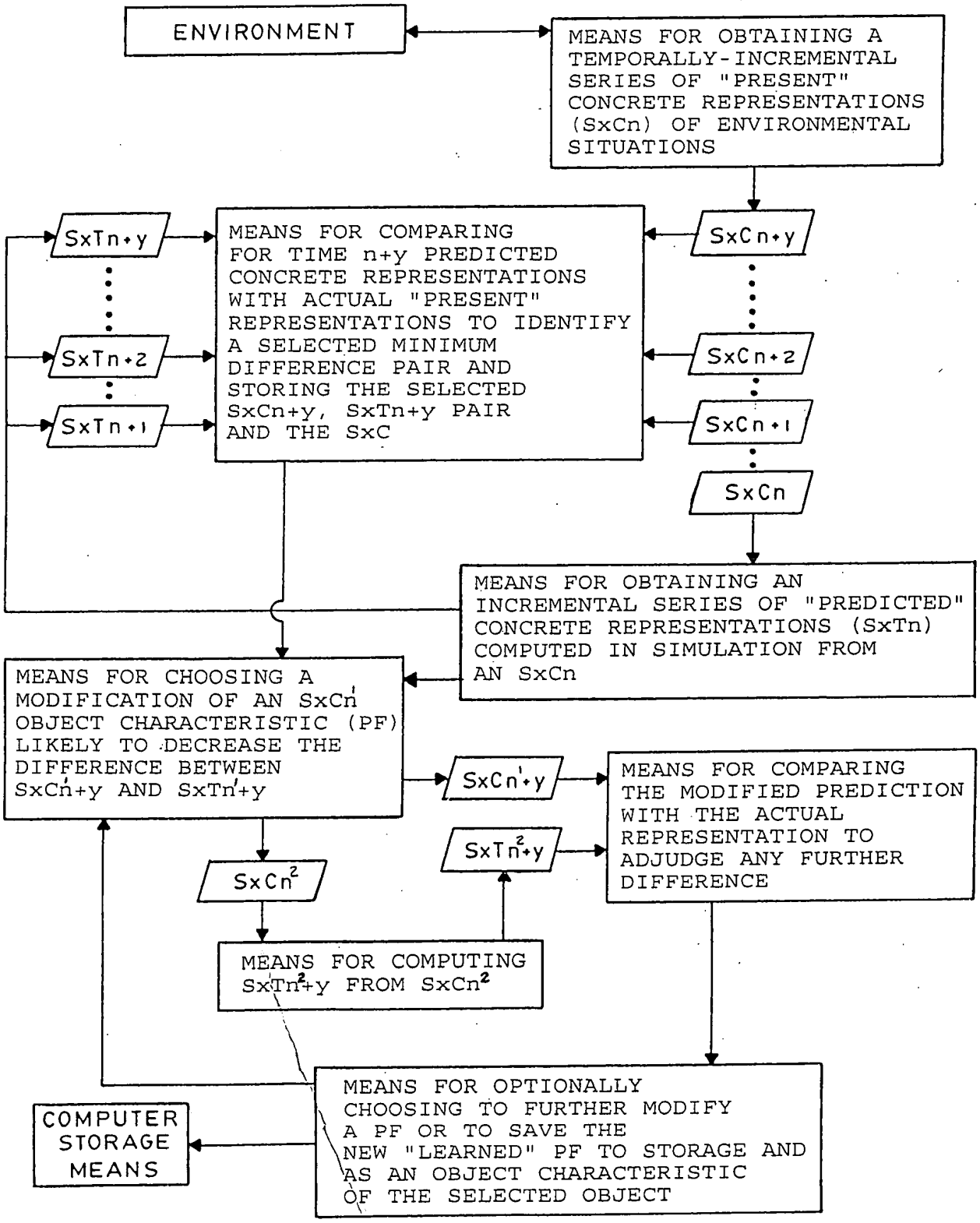


FIG. 10B

STEPS IN A COMPUTER PROGRAM FOR IMPLEMENTING RELEVANCY LEARNING IN AN AUTONOMOUS DECISION SYSTEM

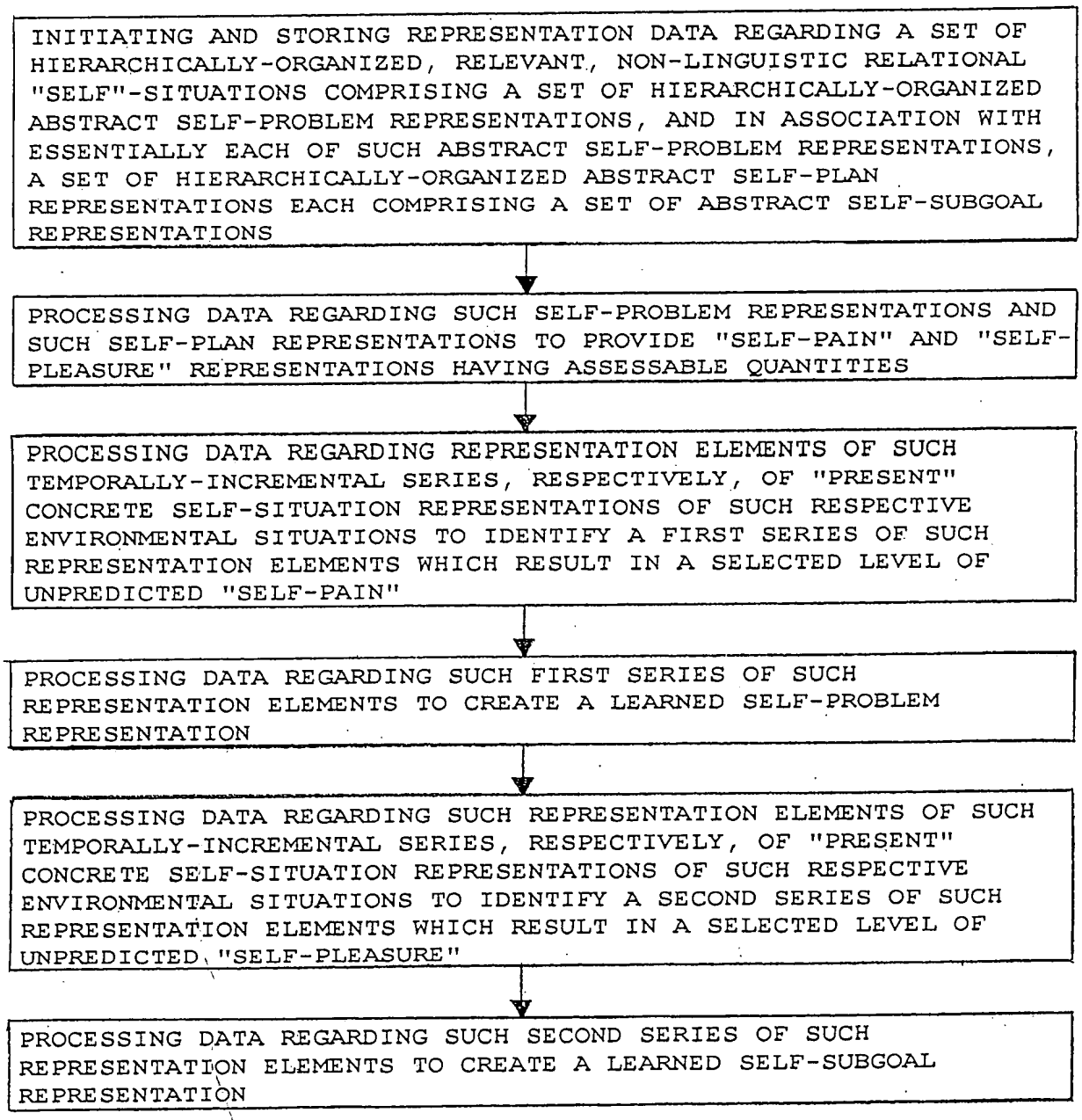


FIG. 11

HIGH LEVEL FLOW CHART OF ENTERTAINMENT SYSTEM

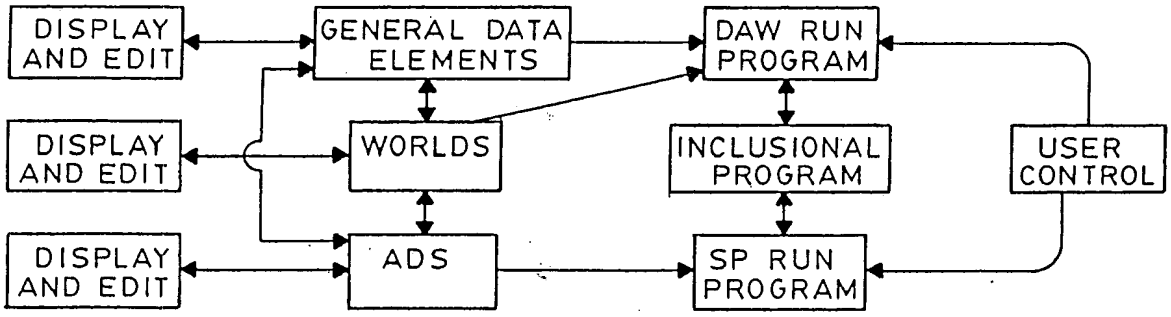


FIG. 12

SUMMARY OF PRIMARY SOFTWARE SYSTEMS AND FUNCTIONS

World:
Data Building/Saving Interfaces
- Writing RF data
- Writing RFM data
- Writing PFM data
- Writing Ri/Pi/ data
- Writing r data
- Writing Si data
- Writing dsd data
- Writing PF data
- Writing SCAN data
- Writing Metabolism Choices

Object Building/Saving Interfaces
- Selecting an Object Name
- Selecting RF's
- Selecting Doings and Animation
- Selecting Meta-PF's

Meta-PF Building/Saving Interfaces
- Writing Meta-PF's

World Building/Saving Interfaces
- Selecting a World Name
- Selecting an Object
- Plunking an Object

World Incremental Simulation

ADS:
Data Building Interfaces
- Writing Problem Nodes: hierarchy, Sip, dD
- Writing Plan Nodes: hierarchy, Sigp, dD, dTd, dc
- Writing Emotional Attributes
- Writing Facial/Emotional Attributes
- Writing SCAN modes
- Writing Questions
- Writing Story/Answer Snippets
- Writing Story/Answer Synonyms

Selecting ADS Attributes
- Selecting IQ
- Selecting Emotion/Face Attributes
- Selecting Story Start/Stop Attributes

Overall ADS-Run Systems
- Sensing What Situation
- Recognizing What Situation
- Deciding What Problem Node
- Deciding What Plan Node
- Look-For System to SCAN
- Emotional/Facial System
- Story-Telling System
- Question-Answering System
- Predicting Threats and Opportunities
- Making Action Decisions

Run:
Selecting Screens To Show: World, Radar, Metabolism, Face
Selecting a World
Plunking an ADS
Selecting Tell/Story Modes
Selecting Run Modes
Stopping and Starting

FIG. 13

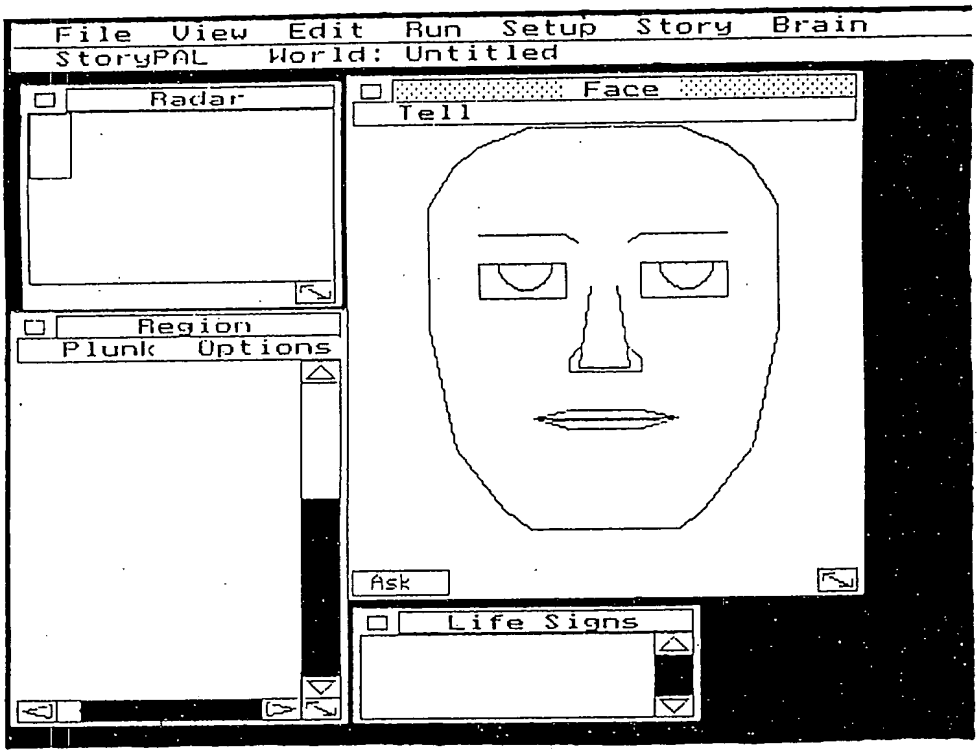


FIG. 14

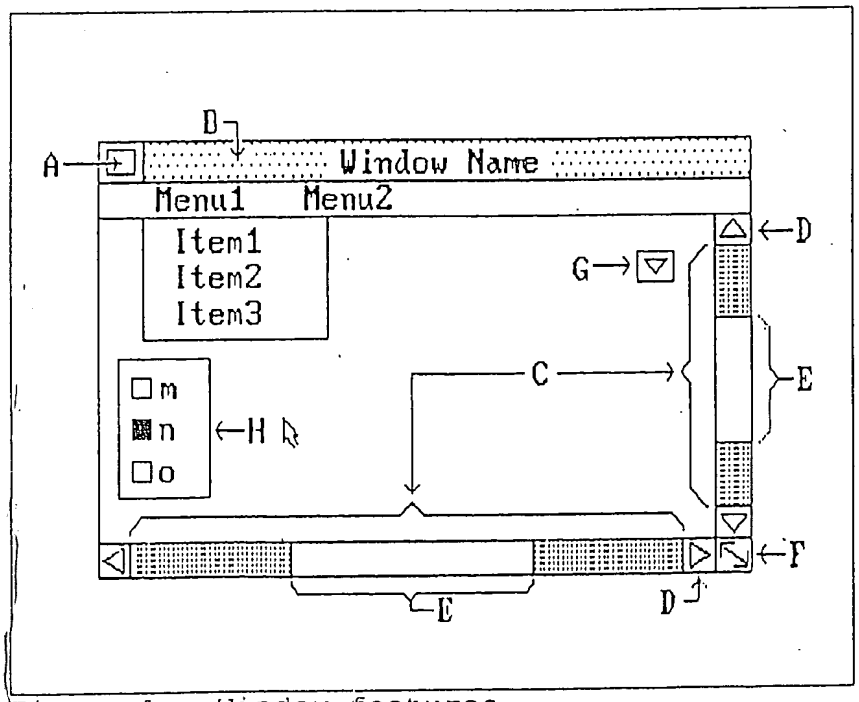


Figure 1: Window features

FIG. 15

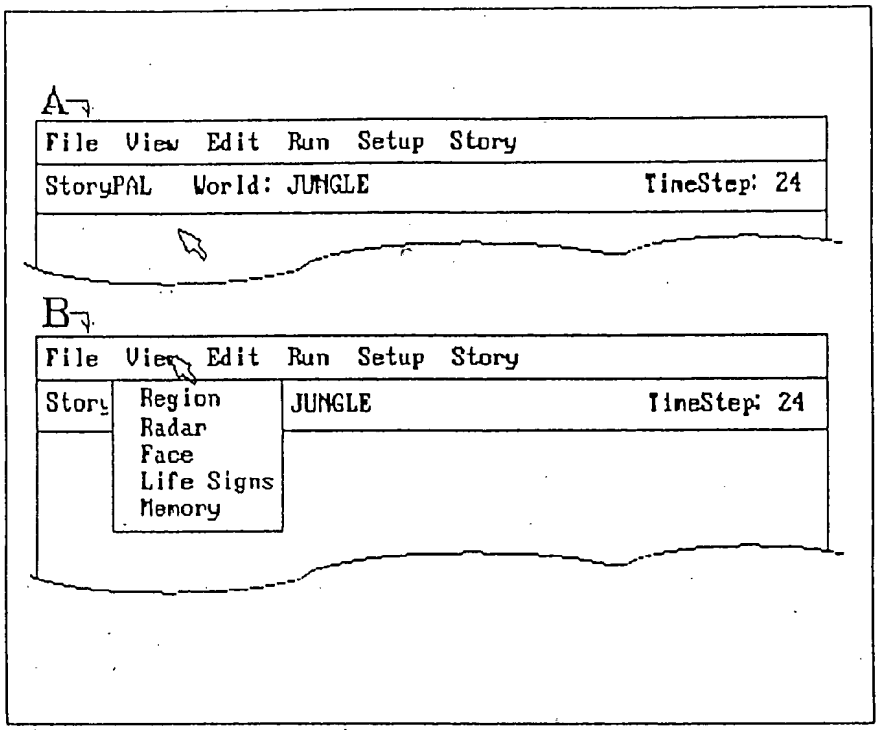


Figure 2: The Main Screen: A - No menus open, and B - the View menu open

FIG. 16

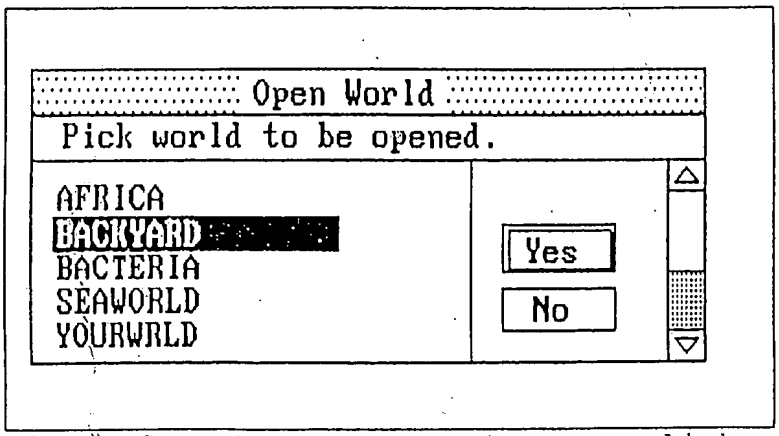


Figure 3: What you see when you click on Open

FIG. 17

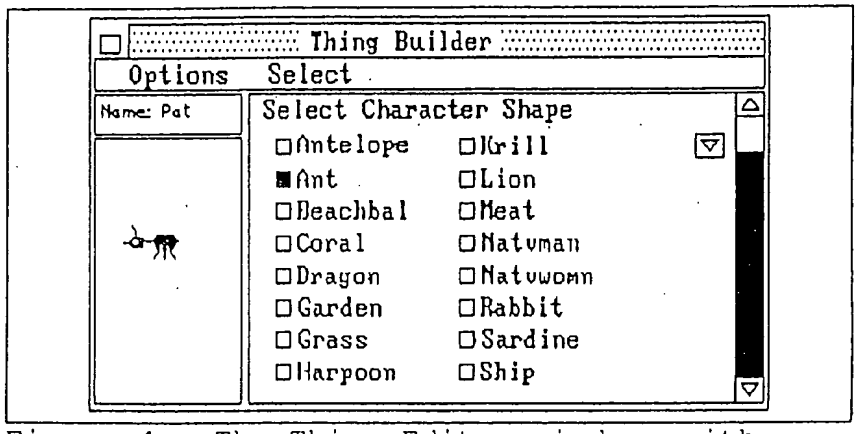


Figure 4: The Thing Editor window, with several sub-windows

FIG. 18

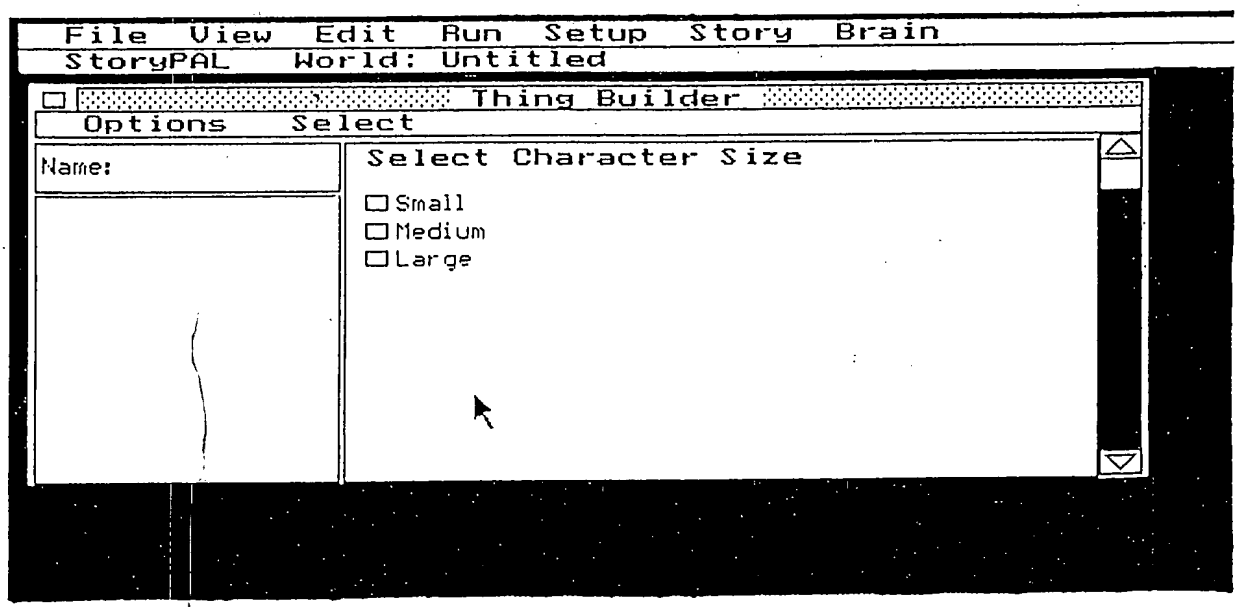


FIG. 19

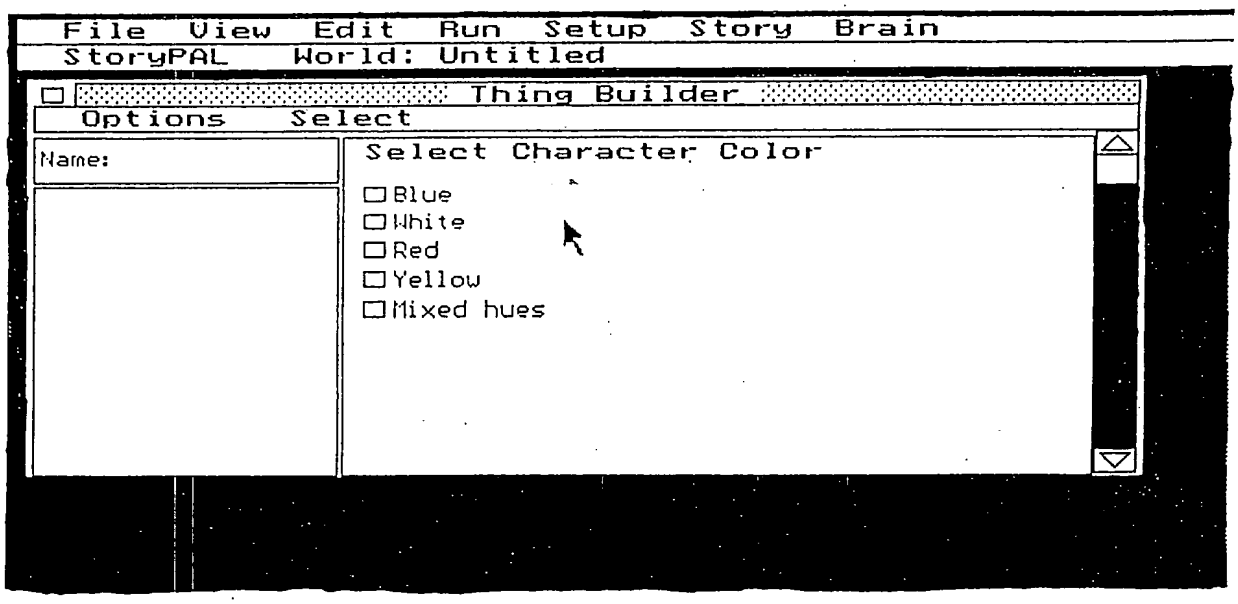

File View Edit Run Setup Story Brain
StoryPAL World: Untitled
Thing Builder
Options Select
Name:
Select Character Color
Blue
White
Red
Yellow
Mixed hues

FIG. 20

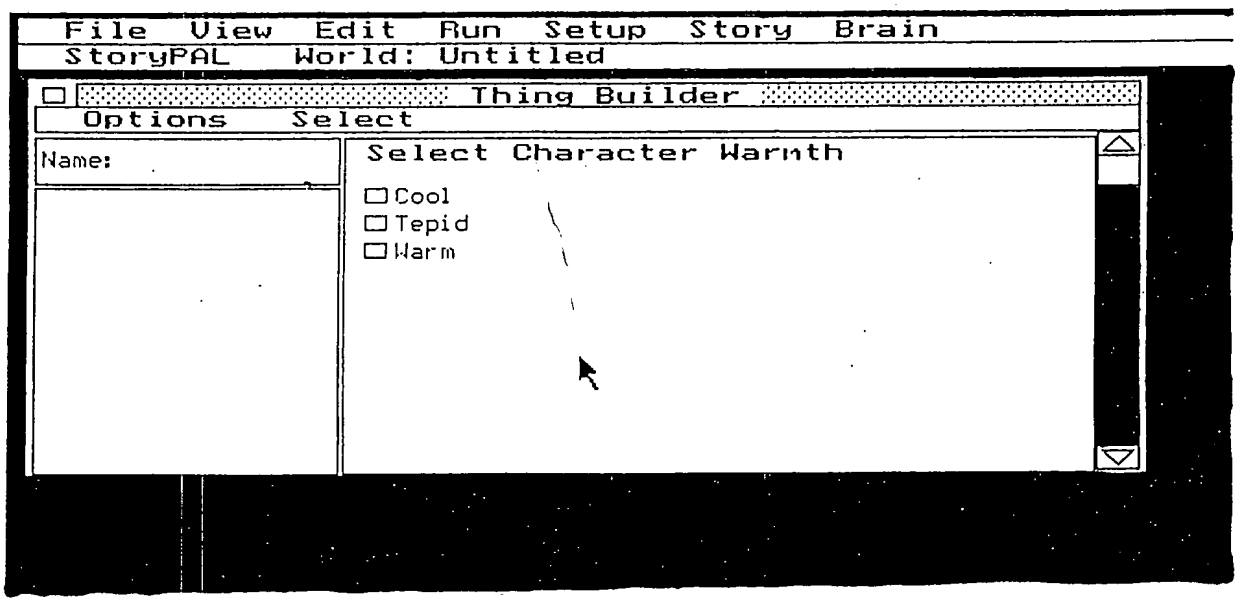

File View Edit Run Setup Story Brain
StoryPAL World: Untitled
Thing Builder
Options Select
Name:
Select Character Warmth
Cool
Tepid
Warm

FIG. 21

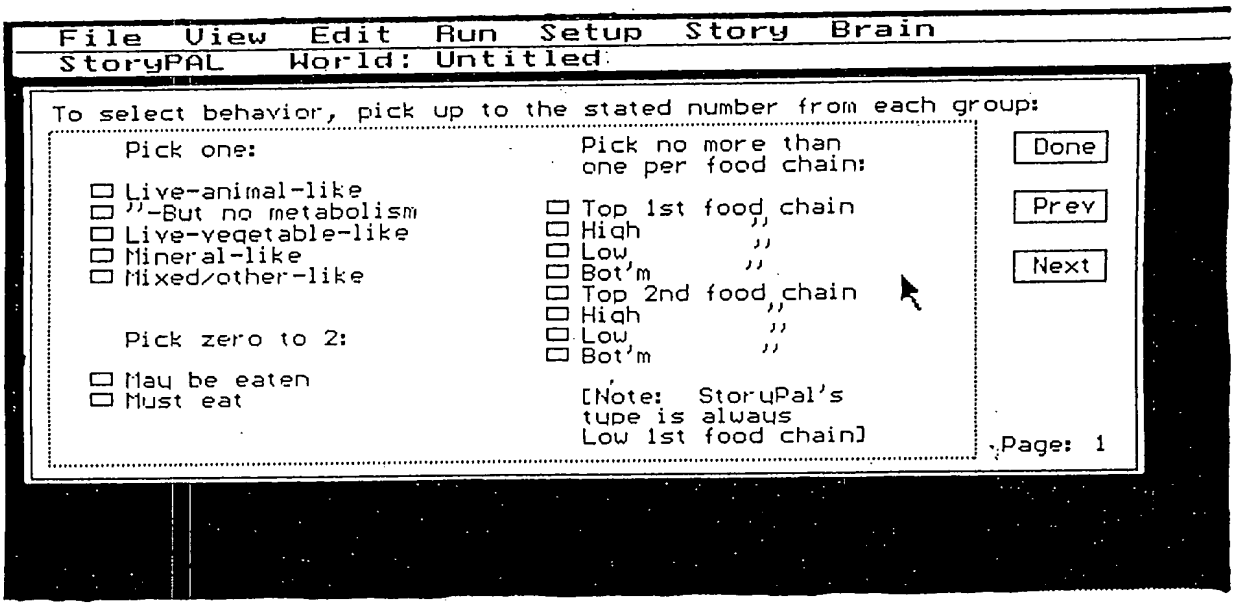

File View Edit Run Setup Story Brain
StoryPAL World: Untitled
To select behavior, pick up to the stated number from each group:
Pick one:
Live-animal-like
"-But no metabolism
Live-vegetable-like
Mineral-like
Mixed/other-like
Pick zero to 2:
May be eaten
Must eat
Pick no more than one per food chain:
Top 1st food chain
High "
Low "
Bot'm "
Top 2nd food chain
High "
Low "
Bot'm "
[Note: StoryPal's type is always Low 1st food chain]
Done
Prev
Next
Page: 1

FIG. 22

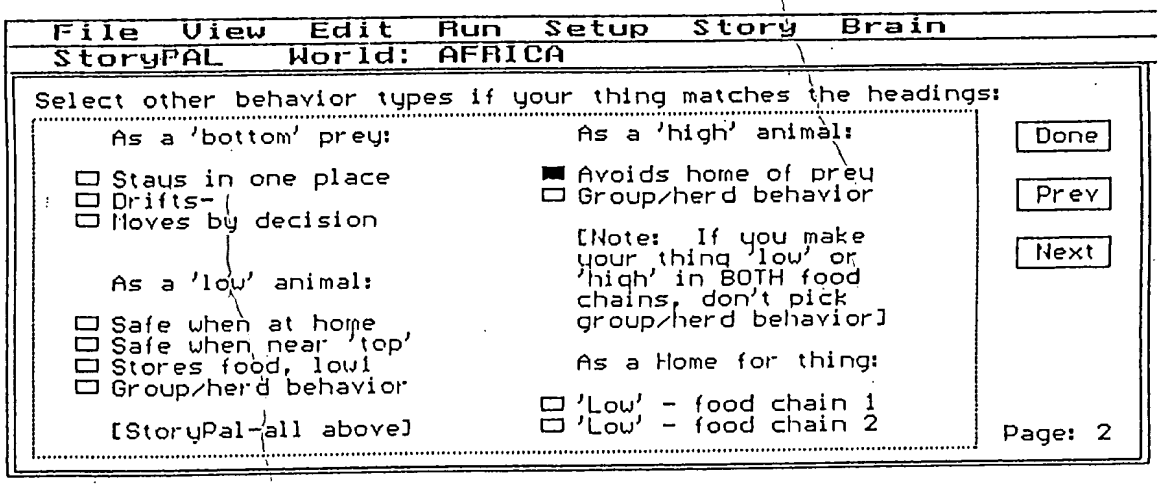

File View Edit Run Setup Story Brain
StoryPAL World: AFRICA
Select other behavior types if your thing matches the headings:
As a 'bottom' prey:
Stays in one place
Drifts-
Moves by decision
As a 'low' animal:
Safe when at home
Safe when near 'top'
Stores food, low1
Group/herd behavior
[StoryPal-all above]
As a 'high' animal:
Avoids home of prey
Group/herd behavior
[Note: If you make your thing 'low' or 'high' in BOTH food chains, don't pick group/herd behavior]
As a Home for thing:
'Low' - food chain 1
'Low' - food chain 2
Done
Prev
Next
Page: 2

FIG. 23

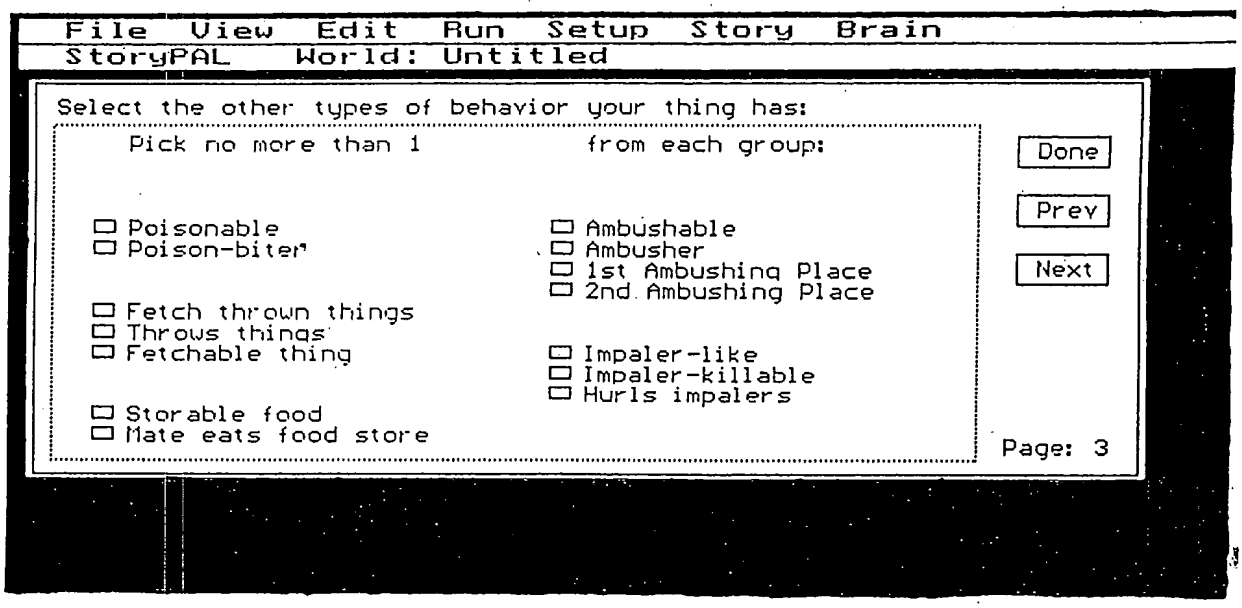

File View Edit Run Setup Story Brain
StoryPAL World: Untitled
Select the other types of behavior your thing has:
Pick no more than 1 from each group:
Done
Prev
Next
Poisonable
Poison-biter
Ambushable
Ambusher
1st Ambushing Place
2nd Ambushing Place
Fetch thrown things
Throws things
Fetchable thing
Impaler-like
Impaler-killable
Hurls impalers
Storable food
Mate eats food store
Page: 3

FIG. 24

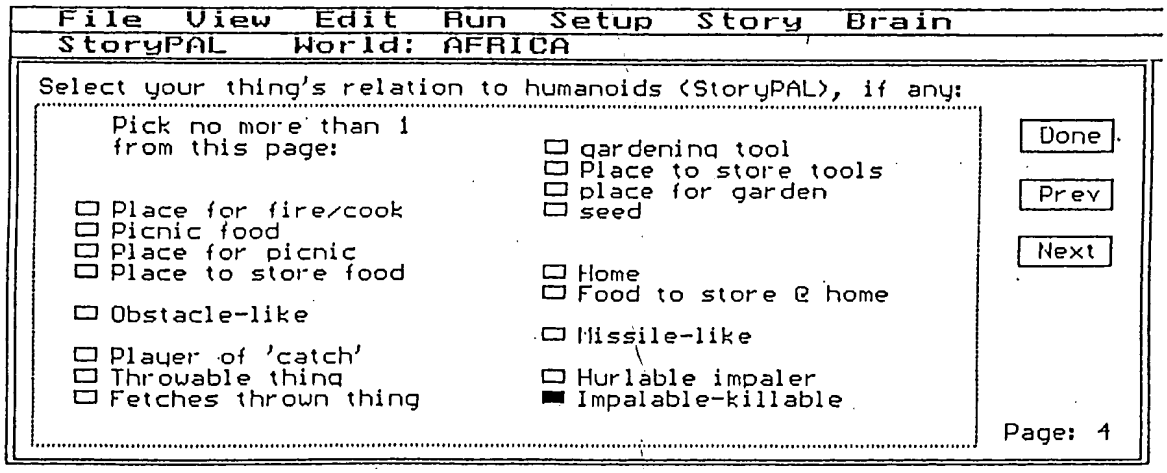

File View Edit Run Setup Story Brain
StoryPAL World: AFRICA
Select your thing's relation to humanoids (StoryPAL), if any:
Pick no more than 1 from this page:
Done
Prev
Next
gardening tool
Place to store tools
place for garden
seed
Place for fire/cook
Picnic food
Place for picnic
Place to store food
Home
Food to store @ home
Obstacle-like
Missile-like
Player of 'catch'
Throwable thing
Fetches thrown thing
Hurlable impaler
Impalable-killable
Page: 4

FIG. 25

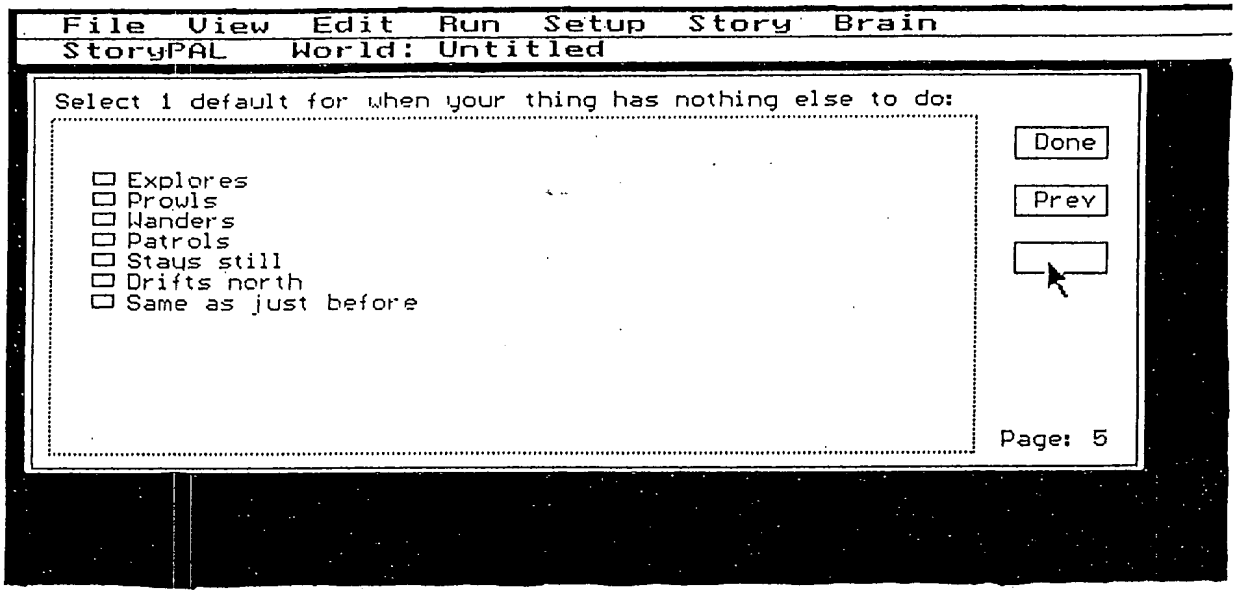


FIG. 26

IQ Setting

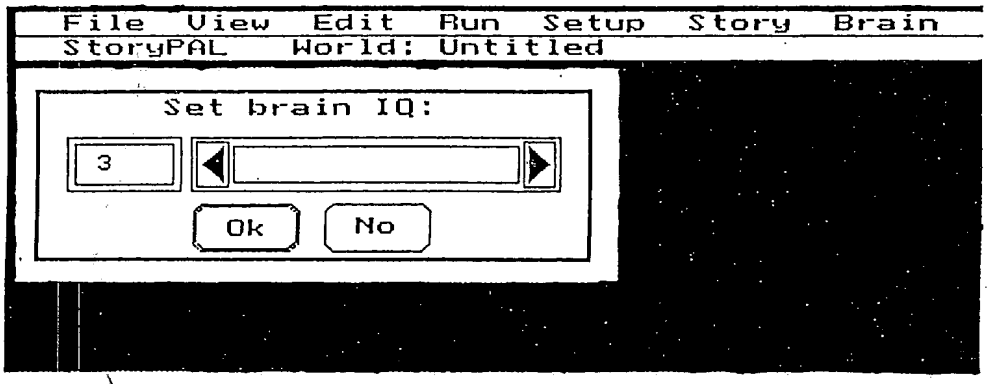


FIG. 27

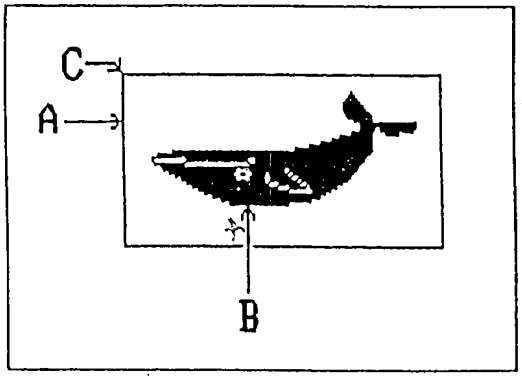

C
A
B

FIG. 28

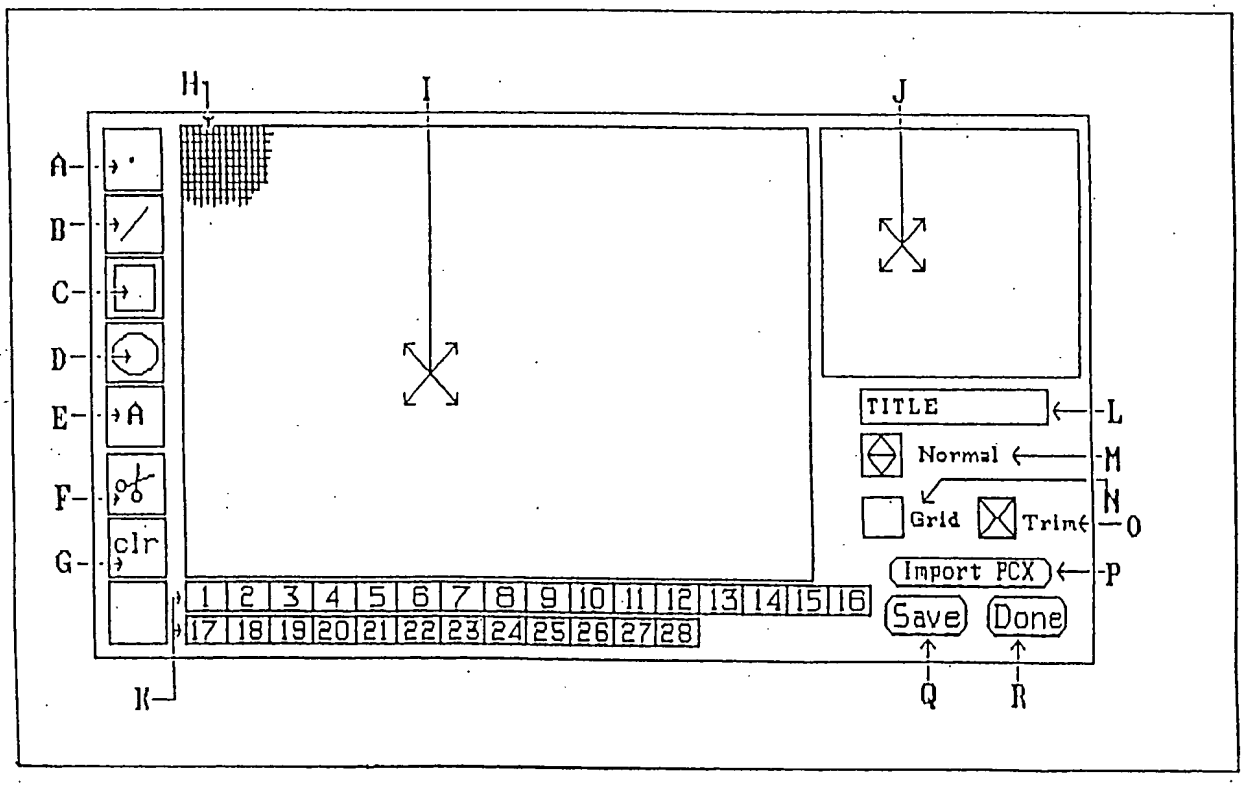

H
I
J
A
B
C
D
E
F
G
clr
TITLE
L
Normal
M
Grid
Trim
N
O
Import PCX
P
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
17 18 19 20 21 22 23 24 25 26 27 28
Save
Done
K
Q
R

FIG. 29

The Expert System Main Screen Menus

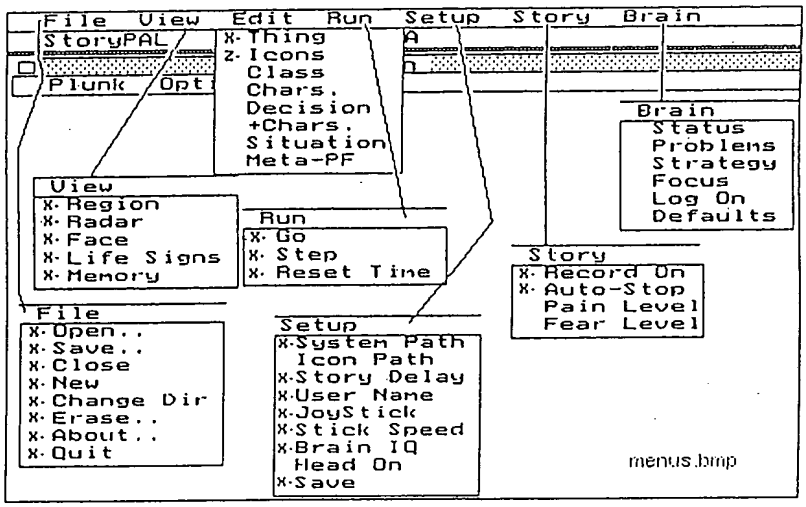
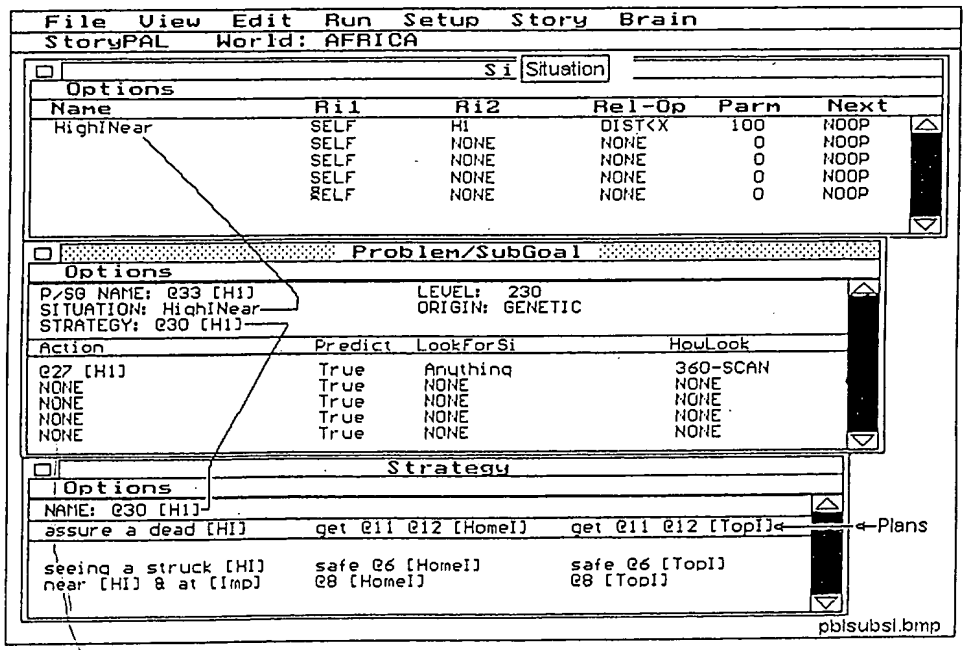


FIG. 31



```
[Get_Next_SX] - FantasyTime False
[Deep Thought] - Ctime: 1
Scanning strategy find something to do rfs:
  No rfs present.
Scanning @14 rfs:
  No rfs present.

SxC(1):
RxName      X   Y  Decision              Speed Rads  Reason  Situation
-------------------------------------------------------------------------
StoryPAL   232 214  NIL                    0     1.67  MARV    NIL
lion       243 254  race to LowI           0     2.62  RF-SCAN LowINear
tree       108  50  don't move             0     6.28  RF-SCAN Anything
native     269  10  wander                 0     1.58  RF-SCAN Anything
rabbit       2 259  explore                0     4.43  RF-SCAN Anything
rabbit     257 111  explore                0     5.99  RF-SCAN Anything
deer       441 314  patrol                 0     1.84  RF-SCAN Anything
grass       29 326  don't move             0     5.26  RF-SCAN Anything
grass      323 347  don't move             0     0.38  RF-SCAN Anything
grass      335  86  don't move             0     5.83  RF-SCAN Anything

[What_Now_Porpoise] : FantasyTime False
  Problem: @33 [H1]
      Row: 1
      Dsd: @27 [H1]              180 20  2H1
      Reg: TIRED
    Value: 10
[Set_New_Problem]
[Find_Strategy]
[Find_Starting_Place_In_Strategy] - FantasyTime False
  SUB-P/SG NAME: seeing a struck [HI]
  SUB-P/SG NAME: safe @6 [HomeI]
  SUB-P/SG NAME: safe @6 [TopI]
  SUB-P/SG NAME: near [HI] & at [Imp]
  SUB-P/SG NAME: @8 [HomeI]
  SUB-P/SG NAME: @8 [TopI]
  HIT Strategy
   Strat.:@30 [H1]
     Plan:get @11 @12 [HomeI]
      Row:2
  SubGoal: @8 [HomeI]
      Row: 1

[Strat.Make_Decision]
[Predicting Strategy]  Try: 1 of 25
  Trial DSD: @19 @12 [HomeI]
[Get_Next_SX] - FantasyTime True

Ending SxT(1):
RxName      X   Y   Decision              Speed Rads  Reason   Situation
-------------------------------------------------------------------------
StoryPAL   219 197  @19 @12 [HomeI]       21    2.22  MARV     NIL
lion       236 230  race to LowI          24    1.84  RF-SCAN  LowINear
tree       108  50  don't move            0     6.13  RF-SCAN  Anything
native     268   1  wander                10    4.70  RF-SCAN  Anything
rabbit      11 259  explore               10    6.21  RF-SCAN  Anything
rabbit     264 104  explore               10    0.67  RF-SCAN  Anything
deer       435 292  patrol                22    1.84  RF-SCAN  Anything
grass       29 326  don't move            0     0.67  RF-SCAN  Anything
grass      323 347  don't move            0     1.52  RF-SCAN  Anything
grass      335  86  don't move            0     0.00  RF-SCAN  Anything

[What_Now_Porpoise] : FantasyTime True
[Next_N_Subgoals_Reached] - FantasyTime True
  SUB-P/SG NAME: safe @6 [HomeI]
  No next(n) subgoal found.

[Predicting Strategy]  Try: 2 of 25
  Trial DSD: @19 @12 [HomeI]
[Get_Next_SX] - FantasyTime True
```

FIG. 30A-2

Ending SxT(2):

RxName	X	Y	Decision	Speed	Rads	Reason	Situation
StoryPAL	206	180	@19 @12 [HomeI]	21	2.22	MARV	NIL
lion	225	208	race to LowI	24	2.05	RF-SCAN	LowINear
tree	108	50	don't move	0	6.13	RF-SCAN	Anything
native	267	10	wander	10	4.70	RF-SCAN	Anything
rabbit	13	268	explore	10	5.01	RF-SCAN	Anything
rabbit	273	104	explore	10	6.27	RF-SCAN	Anything
deer	429	270	patrol	22	1.84	RF-SCAN	Anything
grass	29	326	don't move	0	0.67	RF-SCAN	Anything
grass	323	347	don't move	0	1.52	RF-SCAN	Anything
grass	335	86	don't move	0	0.00	RF-SCAN	Anything

[What_Now_Porpoise] : FantasyTime True
[Next_N_Subgoals_Reached] - FantasyTime True
SUB-P/SG NAME: safe @6 [HomeI]
No next(n) subgoal found.

[Predicting Strategy] Try: 3 of 25
Trial DSD: @19 @12 [HomeI]
[Get_Next_SX] - FantasyTime True

Ending SxT(3):

RxName	X	Y	Decision	Speed	Rads	Reason	Situation
StoryPAL	193	163	@19 @12 [HomeI]	21	2.22	MARV	NIL
lion	211	188	race to LowI	24	2.17	RF-SCAN	LowINear
tree	108	50	don't move	0	6.13	RF-SCAN	Anything
native	266	19	wander	10	4.70	RF-SCAN	Anything
rabbit	16	258	explore	10	1.20	RF-SCAN	Anything
rabbit	265	96	explore	10	2.36	RF-SCAN	Anything
deer	423	248	patrol	22	1.84	RF-SCAN	Anything
grass	29	326	don't move	0	0.67	RF-SCAN	Anything
grass	323	347	don't move	0	1.52	RF-SCAN	Anything
grass	335	86	don't move	0	0.00	RF-SCAN	Anything

[What_Now_Porpoise] : FantasyTime True
[Next_N_Subgoals_Reached] - FantasyTime True
SUB-P/SG NAME: safe @6 [HomeI]

No next(n) subgoal found.

[Predicting Strategy] Try: 4 of 25
Trial DSD: @19 @12 [HomeI]
[Get_Next_SX] - FantasyTime True

Ending SxT(4):

RxName	X	Y	Decision	Speed	Rads	Reason	Situation
StoryPAL	180	146	@19 @12 [HomeI]	21	2.22	MARV	NIL
lion	196	168	race to LowI	24	2.19	RF-SCAN	LowINear
tree	108	50	don't move	0	6.13	RF-SCAN	Anything
native	265	28	wander	10	4.70	RF-SCAN	Anything
rabbit	12	248	explore	10	1.96	RF-SCAN	Anything
rabbit	276	85	move from LowI	16	0.75	RF-SCAN	LowINear
deer	417	226	patrol	22	1.84	RF-SCAN	Anything
grass	29	326	don't move	0	0.67	RF-SCAN	Anything
grass	323	347	don't move	0	1.52	RF-SCAN	Anything
grass	335	86	don't move	0	0.00	RF-SCAN	Anything

[What_Now_Porpoise] : FantasyTime True
[Next_N_Subgoals_Reached] - FantasyTime True
SUB-P/SG NAME: safe @6 [HomeI]
No next(n) subgoal found.

[Predicting Strategy] Try: 5 of 25
Trial DSD: @19 @12 [HomeI]
[Get_Next_SX] - FantasyTime True

FIG. 30A-3

Ending SxT(5):

RxName	X	Y	Decision	Speed	Rads	Reason	Situation
StoryPAL	167	129	@19 @12 [HomeI]	21	2.21	MARV	NIL
lion	181	148	race to LowI	24	2.20	RF-SCAN	LowINear
tree	108	50	don't move	0	6.13	RF-SCAN	Anything
native	264	37	wander	10	4.70	RF-SCAN	Anything
rabbit	15	238	explore	10	1.22	RF-SCAN	Anything
rabbit	278	94	explore	10	5.01	RF-SCAN	Anything
deer	411	204	patrol	22	1.84	RF-SCAN	Anything
grass	29	326	don't move	0	0.67	RF-SCAN	Anything
grass	323	347	don't move	0	1.52	RF-SCAN	Anything
grass	335	86	don't move	0	0.00	RF-SCAN	Anything

[What_Now_Porpoise] : FantasyTime True
[Next_N_Subgoals_Reached] - FantasyTime True
  SUB-P/SG NAME: safe @6 [HomeI]
  No next(n) subgoal found.

[Predicting Strategy] Try: 6 of 25
  Trial DSD: @19 @12 [HomeI]
[Get_Next_SX] - FantasyTime True

Ending SxT(6):

RxName	X	Y	Decision	Speed	Rads	Reason	Situation
StoryPAL	154	112	@19 @12 [HomeI]	21	2.21	MARV	NIL
lion	166	128	race to LowI	24	2.21	RF-SCAN	LowINear
tree	108	50	don't move	0	6.13	RF-SCAN	Anything
native	263	46	wander	10	4.70	RF-SCAN	Anything
rabbit	24	237	explore	10	0.04	RF-SCAN	Anything
rabbit	277	103	explore	10	4.63	RF-SCAN	Anything
deer	405	182	patrol	22	1.84	RF-SCAN	Anything
grass	29	326	don't move	0	0.67	RF-SCAN	Anything
grass	323	347	don't move	0	1.52	RF-SCAN	Anything
grass	335	86	don't move	0	0.00	RF-SCAN	Anything

[What_Now_Porpoise] : FantasyTime True
  Failed worse because of higher problem: the [H1]'s bite-OUCH
  Strategy prediction Failed.

  (S) LOST PLAN!
  Problem: @33 [H1]
      Row: 1
  SubGoal: @8 [HomeI]
      Row: 1
      Dsd: @27 [H1] 180 20 2H1
      Reg: TIRED
    Value: 10
[Problem.Make_Decision]

StoryPAL registers:

HURT	0	HUNGER	52	TIRED	12	HURTRATE	0
HUNGERRATE	0	TIREDRATE	0	MOTORRATE	58	FEAR	230
SURPRISE	0	DISAPPOINT	230	FRUSTRATED	128	HOPELESS	0
PAIN/JOY	127	TOTAL	207		0		0
	0		0		0		0

SxX(1):

RxName	X	Y	Decision	Speed	Rads	Reason	Situation
lion	236	230	race to LowI	24	1.84	PLUNKED	LowINear
tree	108	50	don't move	0	5.36	PLUNKED	Anything
native	268	1	wander	10	4.70	PLUNKED	Anything
rabbit	671	75	explore	10	5.33	PLUNKED	Anything
rabbit	726	341	explore	10	5.09	PLUNKED	Anything
rabbit	1	254	explore	10	0.47	PLUNKED	Anything
rabbit	261	102	explore	10	1.05	PLUNKED	Anything
deer	435	295	patrol	19	1.84	PLUNKED	Anything
grass	29	326	don't move	0	0.50	PLUNKED	Anything
grass	323	347	don't move	0	2.17	PLUNKED	Anything
grass	758	94	don't move	0	3.37	PLUNKED	Anything

```
grass       335  86  don't move            0     4.03  PLUNKED Anything
StoryPAL    226 193  @27 [H1]              21    1.84  MARV     NIL

[Get_Next_SX] - FantasyTime False
[Deep Thought] - Ctime: 2
Scanning strategy @30 [H1] rfs:
    Warm         HEAT         171   255
    Medium       SIZE         86    170
    Cool         HEAT         1     85
    Mixed hues   GREEN        128   255
    Yellow       GREEN        1     127
Scanning @33 [H1] rfs:
    Warm         HEAT         171   255
    Yellow       GREEN        1     127
    Medium       SIZE         86    170

SxC(2):
RxName      X   Y   Decision              Speed Rads  Reason  Situation
-------------------------------------------------------------------------
StoryPAL    226 193 NIL                   21    1.84  MARV    NIL
lion        236 230 race to LowI          24    1.84  RF-SCAN LowINear
tree        108  50 don't move            0     5.36  RF-SCAN Anything
native      268   1 wander                10    4.70  RF-SCAN Anything
rabbit        1 254 explore                 10    0.47  RF-SCAN Anything
rabbit      261 102 move from LowI        10    1.05  RF-SCAN LowINear
deer        435 295 patrol                19    1.84  RF-SCAN Anything
grass        29 326 don't move            0     0.50  RF-SCAN Anything
grass       323 347 don't move            0     2.17  RF-SCAN Anything
grass       335  86 don't move            0     4.03  RF-SCAN Anything

[What_Now_Porpoise] : FantasyTime False
  Problem: @33 [H1]
      Row: 1
      Dsd: @27 [H1]            180 20  2H1
      Reg: TIRED
    Value: 10
[Find_Strategy]
[Find_Starting_Place_In_Strategy] - FantasyTime False
  SUB-P/SG NAME: seeing a struck [HI]
  SUB-P/SG NAME: safe @6 [HomeI]
  SUB-P/SG NAME: safe @6 [TopI]
  SUB-P/SG NAME: near [HI] & at [Imp]
  SUB-P/SG NAME: @8 [HomeI]
  SUB-P/SG NAME: @8 [TopI]
  HIT Strategy
   Strat.:@30 [H1]
     Plan:get @11 @12 [HomeI]
      Row:2
  SubGoal: @8 [HomeI]
      Row: 1

[Strat.Make_Decision]
[Predicting Strategy]  Try: 1 of 25
  Trial DSD: @19 @12 [HomeI]
[Get_Next_SX] - FantasyTime True

Ending SxT(1):
RxName      X   Y   Decision              Speed Rads  Reason  Situation
-------------------------------------------------------------------------
StoryPAL    212 176 @19 @12 [HomeI]       21    2.26  MARV    NIL
lion        229 206 race to LowI          24    1.83  RF-SCAN LowINear
tree        108  50 don't move            0     6.13  RF-SCAN Anything
native      267  10 wander                10    4.70  RF-SCAN Anything
rabbit       10 257 explore                10    5.96  RF-SCAN Anything
rabbit      266  87 move from LowI        16    1.20  RF-SCAN LowINear
deer        429 276 patrol                19    1.84  RF-SCAN Anything
grass        29 326 don't move            0     0.67  RF-SCAN Anything
grass       323 347 don't move            0     1.52  RF-SCAN Anything
grass       335  86 don't move            0     0.00  RF-SCAN Anything
```

FIG. 30A-5

```
[What_Now_Porpoise] : FantasyTime True
[Next_N_Subgoals_Reached] - FantasyTime True
  SUB-P/SG NAME: safe @6 [HomeI]
  No next(n) subgoal found.

[Predicting Strategy]  Try: 2 of 25
  Trial DSD: @19 @12 [HomeI]
[Get_Next_SX] - FantasyTime True

Ending SxT(2):
RxName      X   Y   Decision              Speed Rads  Reason  Situation
-------------------------------------------------------------------------
StoryPAL   198 159  @19 @12 [HomeI]       21    2.26  MARV    NIL
lion       217 185  race to LowI          24    2.09  RF-SCAN LowINear
tree       108  50  don't move            0     6.13  RF-SCAN Anything
native     266  19  wander                10    4.70  RF-SCAN Anything
rabbit      19 259  explore               10    5.98  RF-SCAN Anything
rabbit     272  94  explore               10    5.42  RF-SCAN Anything
deer       423 257  patrol                19    1.84  RF-SCAN Anything
grass       29 326  don't move            0     0.67  RF-SCAN Anything
grass      323 347  don't move            0     1.52  RF-SCAN Anything
grass      335  86  don't move            0     0.00  RF-SCAN Anything

[What_Now_Porpoise] : FantasyTime True
[Next_N_Subgoals_Reached] - FantasyTime True
  SUB-P/SG NAME: safe @6 [HomeI]
  No next(n) subgoal found.

[Predicting Strategy]  Try: 3 of 25
  Trial DSD: @19 @12 [HomeI]
[Get_Next_SX] - FantasyTime True

Ending SxT(3):
RxName      X   Y   Decision              Speed Rads  Reason  Situation
-------------------------------------------------------------------------
StoryPAL   184 142  @19 @12 [HomeI]       21    2.26  MARV    NIL
lion       202 165  race to LowI          24    2.20  RF-SCAN LowINear
tree       108  50  don't move            0     6.13  RF-SCAN Anything
native     265  28  wander                10    4.70  RF-SCAN Anything
rabbit      27 253  explore               10    0.60  RF-SCAN Anything
rabbit     284  83  move from LowI        16    0.72  RF-SCAN LowINear
deer       417 238  patrol                19    1.84  RF-SCAN Anything
grass       29 326  don't move            0     0.67  RF-SCAN Anything
grass      323 347  don't move            0     1.52  RF-SCAN Anything
grass      335  86  don't move            0     0.00  RF-SCAN Anything

[What_Now_Porpoise] : FantasyTime True

[Next_N_Subgoals_Reached] - FantasyTime True
  SUB-P/SG NAME: safe @6 [HomeI]
  No next(n) subgoal found.

[Predicting Strategy]  Try: 4 of 25
  Trial DSD: @19 @12 [HomeI]
[Get_Next_SX] - FantasyTime True
Ending SxT(4):
RxName      X   Y   Decision              Speed Rads  Reason  Situation
-------------------------------------------------------------------------
StoryPAL   170 125  @19 @12 [HomeI]       21    2.26  MARV    NIL
lion       187 146  race to LowI          24    2.23  RF-SCAN LowINear
tree       108  50  don't move            0     6.13  RF-SCAN Anything
native     264  37  wander                10    4.70  RF-SCAN Anything
rabbit      32 261  explore               10    5.25  RF-SCAN Anything
rabbit     274  82  explore               10    3.09  RF-SCAN Anything
deer       411 219  patrol                19    1.84  RF-SCAN Anything
grass       29 326  don't move            0     0.67  RF-SCAN Anything
grass      323 347  don't move            0     1.52  RF-SCAN Anything
grass      335  86  don't move            0     0.00  RF-SCAN Anything
[What_Now_Porpoise] : FantasyTime True
[Next_N_Subgoals_Reached] - FantasyTime True
  SUB-P/SG NAME: safe @6 [HomeI]
  No next(n) subgoal found.
```

FIG. 30A-6

```
[Predicting Strategy]  Try: 5 of 25
  Trial DSD: @19 @12 [HomeI]
[Get_Next_SX] - FantasyTime True

Ending SxT(5):
```

RxName	X	Y	Decision	Speed	Rads	Reason	Situation
StoryPAL	156	108	@19 @12 [HomeI]	21	2.26	MARV	NIL
lion	171	127	race to LowI	24	2.25	RF-SCAN	LowINear
tree	108	50	don't move	0	6.13	RF-SCAN	Anything
native	263	46	wander	10	4.70	RF-SCAN	Anything
rabbit	40	255	explore	10	0.62	RF-SCAN	Anything
rabbit	279	73	explore	10	0.98	RF-SCAN	Anything
deer	405	200	patrol	19	1.84	RF-SCAN	Anything
grass	29	326	don't move	0	0.67	RF-SCAN	Anything
grass	323	347	don't move	0	1.52	RF-SCAN	Anything
grass	335	86	don't move	0	0.00	RF-SCAN	Anything

```
[What_Now_Porpoise] : FantasyTime True
[Next_N_Subgoals_Reached] - FantasyTime True
  SUB-P/SG NAME: safe @6 [HomeI]
  No next(n) subgoal found.

[Predicting Strategy]  Try: 6 of 25
  Trial DSD: @19 @12 [HomeI]
[Get_Next_SX] - FantasyTime True

Ending SxT(6):
```

RxName	X	Y	Decision	Speed	Rads	Reason	Situation
StoryPAL	142	91	@19 @12 [HomeI]	21	2.26	MARV	NIL
lion	156	108	race to LowI	24	2.24	RF-SCAN	LowINear
tree	108	50	don't move	0	6.13	RF-SCAN	Anything
native	262	55	wander	10	4.70	RF-SCAN	Anything
rabbit	30	259	explore	10	3.56	RF-SCAN	Anything
rabbit	284	64	explore	10	0.94	RF-SCAN	Anything
deer	399	181	patrol	19	1.84	RF-SCAN	Anything
grass	29	326	don't move	0	0.67	RF-SCAN	Anything
grass	323	347	don't move	0	1.52	RF-SCAN	Anything
grass	335	86	don't move	0	0.00	RF-SCAN	Anything

```
[What_Now_Porpoise] : FantasyTime True
[Next_N_Subgoals_Reached] - FantasyTime True
  SUB-P/SG NAME: safe @6 [HomeI]
  No next(n) subgoal found.
```

FIG. 32

SPACIO-TEMPORAL RELATIONSHIPS r
IN SOFTWARE EMBODIMENT

Relational Operator	Meaning
NONE	This is the last line. Don't read any further.
DIST>X	The distance between Pi1 (usually the self-object, Ris) and Pi2 is greater than X
DIST<X	The distance between Pi1 and Pi2 is less than X
FASTER	Pi1 is moving faster (more units of distance per Current Situation) than Pi2
SLOWER	Pi1 is moving slower (less units of distance per Current Situation) than Pi2
TOWARD2	Pi1 is moving toward Pi2 ("toward" = Radians in the Parm field. If the angle of movement is less than the Parm field's radians, Toward is True. Zero radians is East or to the Pight on the screen)
TOWARD1	Pi2 is moving toward Pi1
AWAYFR2	Pi1 is moving away from Pi2 (parenthetical note on TOWARD2 applies here)
AWAYFR1	Pi2 is moving away from Pi1
>ANGX	The angle between Pi1 and Pi2 exceeds X (X is in radians)
<ANGX	The angle between Pi1 and Pi2 is less than X (X is in radians)
MOVE>X	Pi1 is moving faster than X
<LOCX	Pi1 is at a location whose screen X-coordinate is less than X
>LOCX	Pi1 is at a location whose screen X-coordinate is greater than X
<LOCY	Pi1 is at a location whose screen Y-coordinate is less than Y
>LOCY	Pi1 is at a location whose screen Y-coordinate is greater than Y
1CL2BYX	Pi1 is closer to Pi2 than X units of distance
1FR2CYX	Pi1 is farther from Pi2 than X units of distance
FEEL	The self Pi is feeling as per what is here set out

FIG. 33

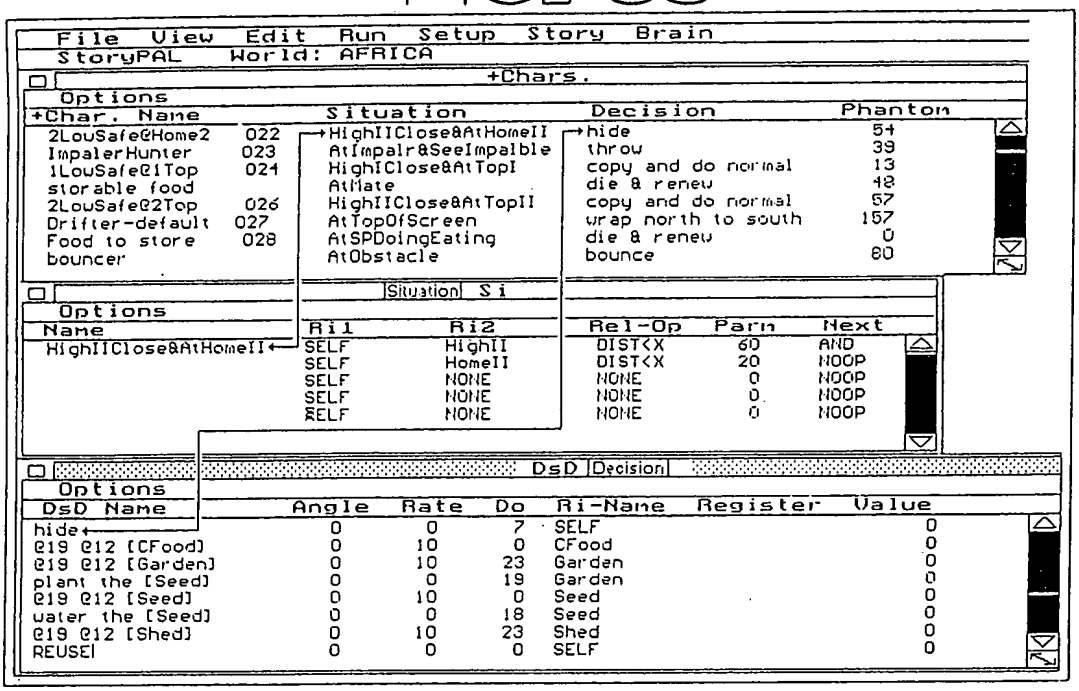


FIG. 34

Do-ings Table

Doing Number	Doing Description	Doing Number	Doing Description
-10	Move the same way as the "Hit" Ri this time or this current situation. Used for "copying" some other Thing, as in flocking behavior.	8	Bitten
-9	Make a buzzing sound	9	Pounce
-8	Make a beeping sound	10	Throw
-7	Make random, or randomize a value.	11	Catch
-6	When Things move off the top of the screen, bring them back to the bottom of the screen.	12	Fly/Swim
-5	Move absolute. A fixed compass setting as opposed to some degrees relative to something.	13	Satiated
-4	Default (On the last page of the Thing Builder is the Thing's Default action)	14	Dead
-3	Be dead	15	Hide/Sit
-2	Move the same as the "Hit" Ri.	16	Light Fire
-1	Do what you did the last move (same direction and rate)	17	Burn
0	Normal (This and the following Do-ings)	18	Douse Fire
1	Chase	19	Reach
2	Flee	20	Door Open
3	Hungry	21	Table Set
4	Hurt	22	Sit & Eat
5	Tired	23	Carry
6	Bite/Eat	24	Cooked
7	Rest	25	Cloaked (is there and can interact and be interacted with, but is invisible)
		26	Using a tool

FIG. 35

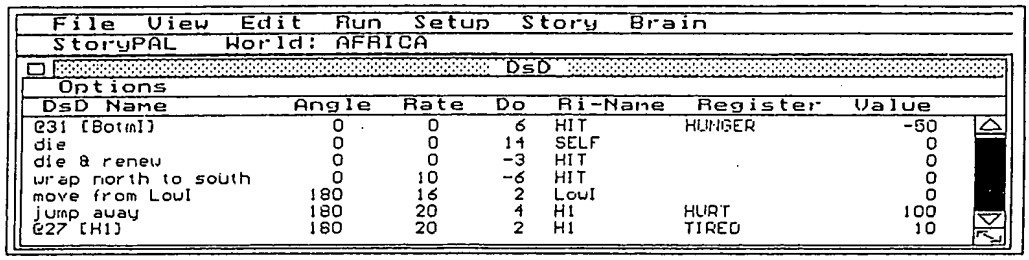


FIG. 36

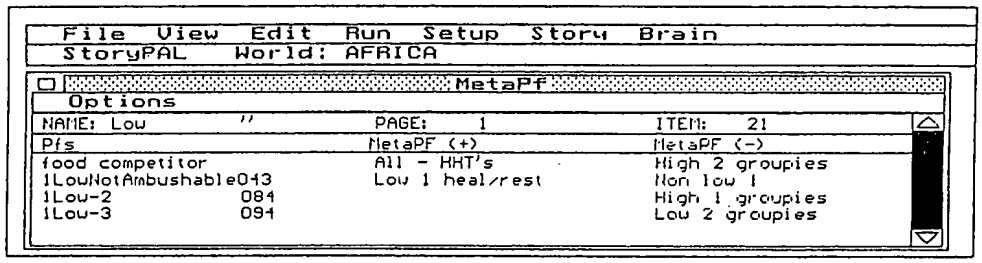


FIG. 37

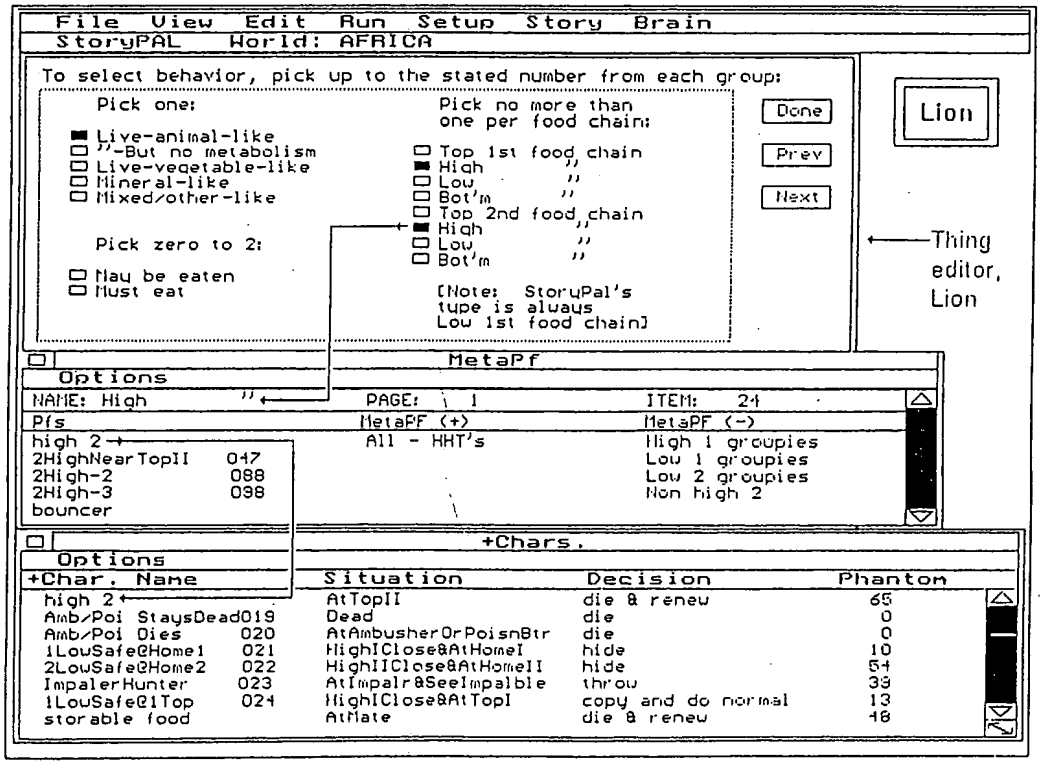


FIG. 38

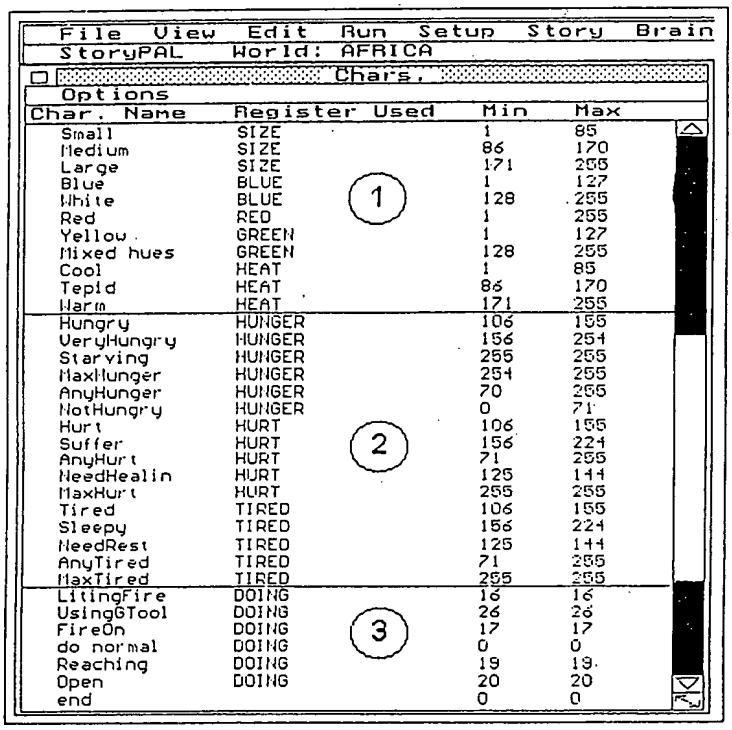


FIG. 39

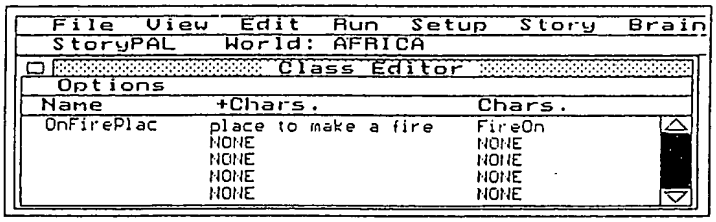


FIG. 40

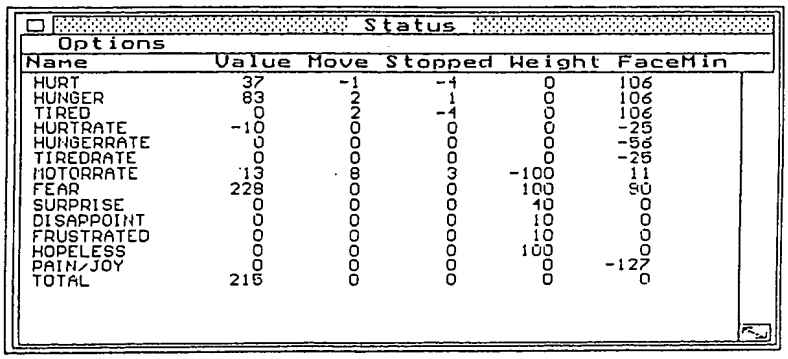


FIG. 41

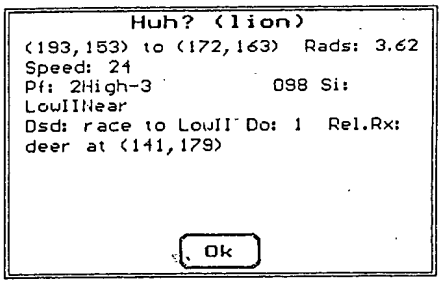


FIG. 42

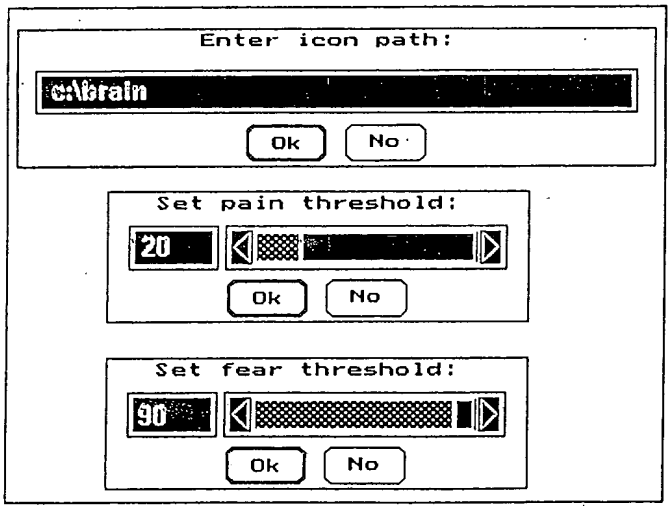


FIG. 43

Data Structure	Created In Window...	Found In Menu...	Used In Other Data Structure Window(s)...
Pi or Ri (Class)	Class	Edit	Situation & Decision
RF or Rf (Char)	Chars.	Edit	Class
PF or Pf (+Char)	+Chars.	Edit	Class & MetaPf
Si (Situation-abstracted)	Situation	Edit	+Char, Problem/Subgoal & Strategy
Problem/Subgoal	Problems	Brain	It isn't
Strategy & Plans	Strategy	Brain	Problem/Subgoal
Do-ing	Coded in the program	none	Decision
DsD (Decision)	Decision	Edit	+Char & Problem/Subgoal
MetaPf	Meta-Pf	Edit	It isn't
Focus	Focus	Brain	Problem/Subgoal

FIG. 44A-1

STEPS IN A COMPUTER PROGRAM FOR IMPLEMENTING EMOTIONAL EXPRESSION IN AN AUTONOMOUS DECISION SYSTEM

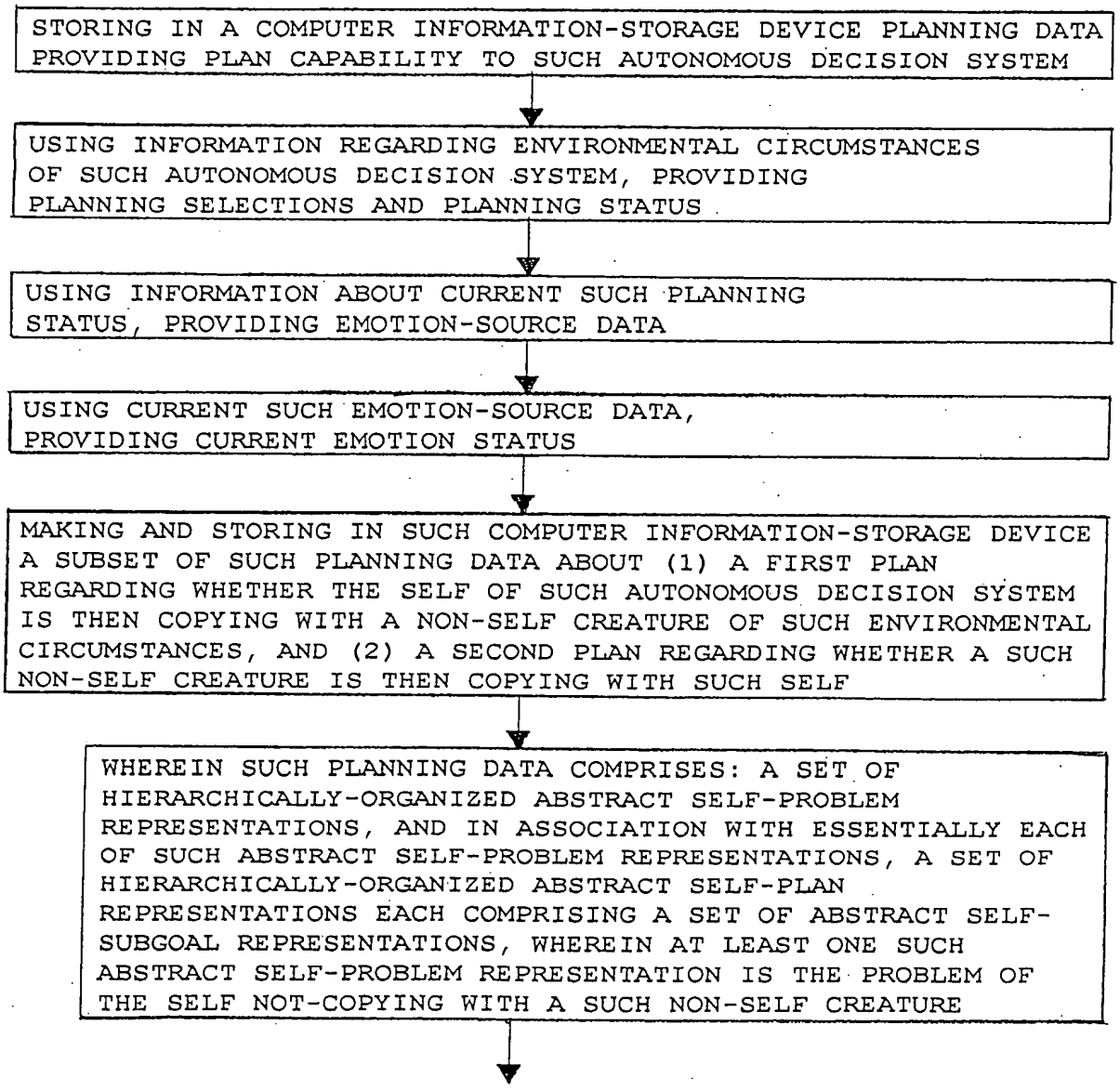


FIG. 44A-2

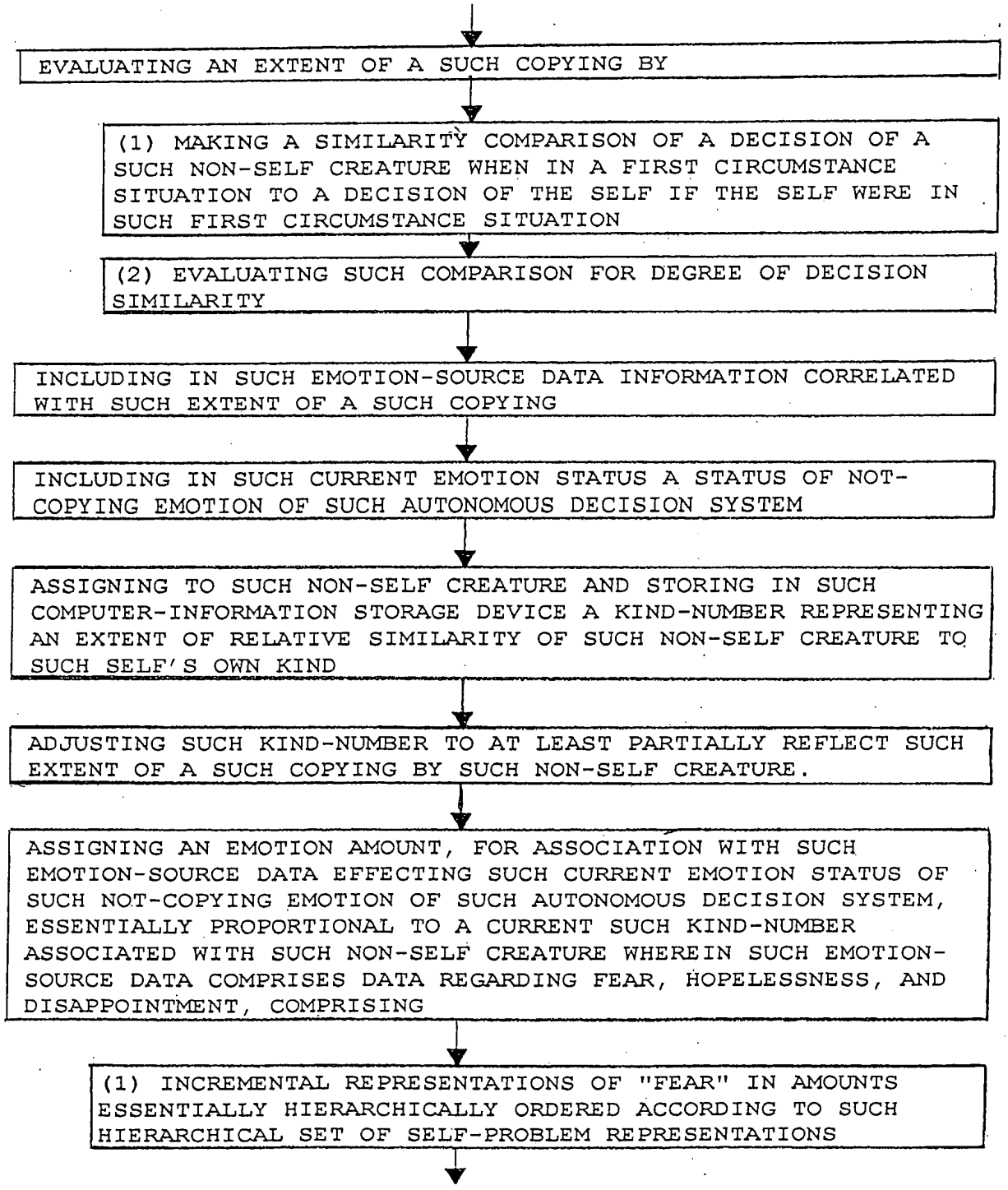
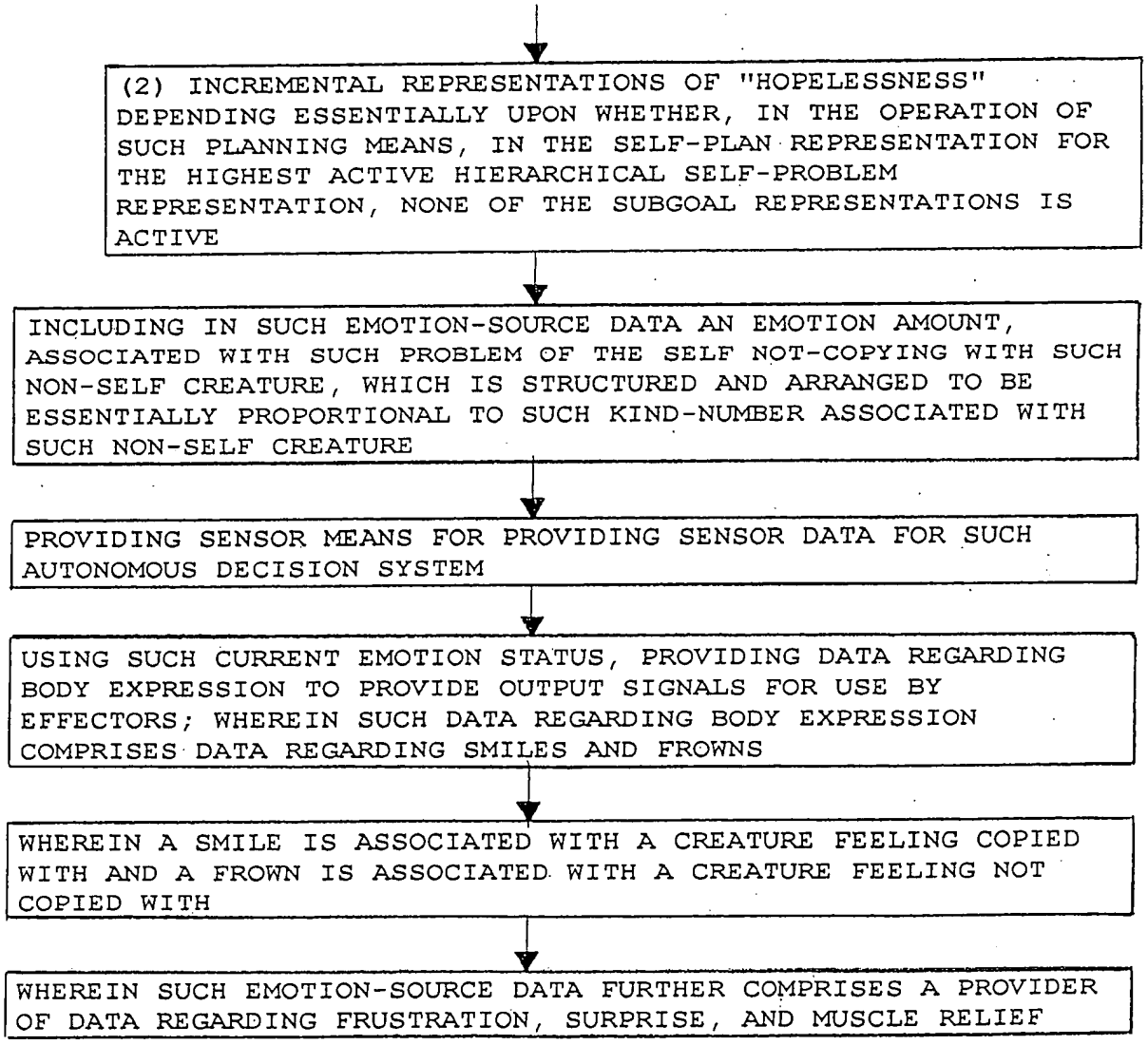

EVALUATING AN EXTENT OF A SUCH COPYING BY
(1) MAKING A SIMILARITY COMPARISON OF A DECISION OF A SUCH NON-SELF CREATURE WHEN IN A FIRST CIRCUMSTANCE SITUATION TO A DECISION OF THE SELF IF THE SELF WERE IN SUCH FIRST CIRCUMSTANCE SITUATION
(2) EVALUATING SUCH COMPARISON FOR DEGREE OF DECISION SIMILARITY
INCLUDING IN SUCH EMOTION-SOURCE DATA INFORMATION CORRELATED WITH SUCH EXTENT OF A SUCH COPYING
INCLUDING IN SUCH CURRENT EMOTION STATUS A STATUS OF NOT-COPYING EMOTION OF SUCH AUTONOMOUS DECISION SYSTEM
ASSIGNING TO SUCH NON-SELF CREATURE AND STORING IN SUCH COMPUTER-INFORMATION STORAGE DEVICE A KIND-NUMBER REPRESENTING AN EXTENT OF RELATIVE SIMILARITY OF SUCH NON-SELF CREATURE TO SUCH SELF'S OWN KIND
ADJUSTING SUCH KIND-NUMBER TO AT LEAST PARTIALLY REFLECT SUCH EXTENT OF A SUCH COPYING BY SUCH NON-SELF CREATURE.
ASSIGNING AN EMOTION AMOUNT, FOR ASSOCIATION WITH SUCH EMOTION-SOURCE DATA EFFECTING SUCH CURRENT EMOTION STATUS OF SUCH NOT-COPYING EMOTION OF SUCH AUTONOMOUS DECISION SYSTEM, ESSENTIALLY PROPORTIONAL TO A CURRENT SUCH KIND-NUMBER ASSOCIATED WITH SUCH NON-SELF CREATURE WHEREIN SUCH EMOTION-SOURCE DATA COMPRISES DATA REGARDING FEAR, HOPELESSNESS, AND DISAPPOINTMENT, COMPRISING
(1) INCREMENTAL REPRESENTATIONS OF "FEAR" IN AMOUNTS ESSENTIALLY HIERARCHICALLY ORDERED ACCORDING TO SUCH HIERARCHICAL SET OF SELF-PROBLEM REPRESENTATIONS

FIG. 44A-3


(2) INCREMENTAL REPRESENTATIONS OF "HOPELESSNESS" DEPENDING ESSENTIALLY UPON WHETHER, IN THE OPERATION OF SUCH PLANNING MEANS, IN THE SELF-PLAN REPRESENTATION FOR THE HIGHEST ACTIVE HIERARCHICAL SELF-PROBLEM REPRESENTATION, NONE OF THE SUBGOAL REPRESENTATIONS IS ACTIVE
INCLUDING IN SUCH EMOTION-SOURCE DATA AN EMOTION AMOUNT, ASSOCIATED WITH SUCH PROBLEM OF THE SELF NOT-COPYING WITH SUCH NON-SELF CREATURE, WHICH IS STRUCTURED AND ARRANGED TO BE ESSENTIALLY PROPORTIONAL TO SUCH KIND-NUMBER ASSOCIATED WITH SUCH NON-SELF CREATURE
PROVIDING SENSOR MEANS FOR PROVIDING SENSOR DATA FOR SUCH AUTONOMOUS DECISION SYSTEM
USING SUCH CURRENT EMOTION STATUS, PROVIDING DATA REGARDING BODY EXPRESSION TO PROVIDE OUTPUT SIGNALS FOR USE BY EFFECTORS; WHEREIN SUCH DATA REGARDING BODY EXPRESSION COMPRISES DATA REGARDING SMILES AND FROWNS
WHEREIN A SMILE IS ASSOCIATED WITH A CREATURE FEELING COPIED WITH AND A FROWN IS ASSOCIATED WITH A CREATURE FEELING NOT COPIED WITH
WHEREIN SUCH EMOTION-SOURCE DATA FURTHER COMPRISES A PROVIDER OF DATA REGARDING FRUSTRATION, SURPRISE, AND MUSCLE RELIEF